CMAQ v4.6 Operational Guidance Document

CMAQ v4.6 Operational Guidance Document

Table of Contents

1. Introduction 1
    1.1. Background and Goals 1
    1.2. Overview of CMAQ System Components 3
    1.3. Features of CMAQ for Application Users 6
    1.4. Features of CMAQ for Air Quality Model Developers 6
    1.5. New Features in Version 4.6 7
    1.6. About this manual 9
    1.7. References 9
2. Overview of the Science in the CMAQ Modeling System 11
    2.1. Features to Achieve the Goals of CMAQ 12
    2.2. CMAQ Input Processors 14
    2.3. CMAQ Chemical Transport Model Science Modules 19
    2.4. The CMAQ User Interface 23
    2.5. References 24
3. CMAQ System Requirements and Installation 27
    3.1. System Recommendations 27
    3.2. Installing CMAQ Source Code 30
    3.3. Compiling CMAQ for the Tutorial Simulation 32
    3.4. Running the CMAQ Tutorial Simulation 34
    3.5. Benchmarking CMAQ 34
4. Input/Output Applications Programming Interface 35
    4.1. Files, Logical Names and Physical Names: 35
    4.2. IO/API Data Structure and Data File Types 36
    4.3. Opening/Creating Data Files in IO/API 38
    4.4. Reading Data Files in IO/API 40
    4.5. Writing Data Files in IO/API 40
    4.6. CMAQ-Related I/O API Utilities 41
    4.7. References 46
5. CMAQ Programs and Libraries 47
    5.1. Overview 47
    5.2. BCON 49
    5.3. CCTM 55
    5.4. CHEMMECH 66
    5.5. ICON 69
    5.6. JPROC 74
    5.7. MCIP 78
    5.8. PARIO 82
    5.9. PDM 83
    5.10. PROCAN 88
    5.11. STENEX 95
    5.12. References 96
6. CMAQ FILES 97
    6.1. CMAQ Input Files 97
    6.2. Basic CMAQ Output Files 131
    6.3. Diagnostic and advanced CMAQ Output Files 133
7. Developing Meteorological Fields 135
    7.1. Credits and Disclaimers For Use of MM5 135
    7.2. System Requirements 136
    7.3. Software Languages 136
    7.4. Data Requirements 136
    7.5. Meteorology Model Pre-Processing 138

7.6. Meteorology Model (MM5) 139
7.7. Meteorology Model Post-Processing 142
7.8. Meteorology Model Visualization and Evaluation 142
7.9. Model Training and User Support 143
7.10. Using MM5 with CMAQ 143
7.11. References 143
8. Developing Emissions Fields 147
8.1. Credits and Disclaimers for Use of SMOKE 147
8.2. Obtaining SMOKE Source Code 147
8.3. System Requirements 147
8.4. Software Languages 147
8.5. Data Requirements 147
8.6. Overview of SMOKE Version 2.1 148
8.7. Emissions Model Visualization and Evaluation 151
8.8. Model Training and User Support 151
8.9. References 151
9. Defining, Grids, Layers, Domains, and Chemistry 153
9.1. Supported CMAQ Coordinate Systems 153
9.2. Horizontal Grids 154
9.3. Vertical Layers 155
9.4. Chemical Mechanism 156
9.5. References 158

List of Figures

2.1. CMAQ Modeling Framework 12
2.2. Chemical Transport Model (CCTM) and preprocessors 14
2.3. Meteorology preprocessing for CMAQ with MCIP 15
2.4. Initial and boundary conditions preprocessing for CMAQ 16
2.5. Clear sky photolysis rate preprocessing for CMAQ 17
2.6. Invoking new/modified gas-phase chemical mechanisms in CMAQ 17
2.7. Plume in Grid treatment in the CMAQ modeling system 18
2.8. Process analysis implementation in the CMAQ modeling system 19
2.9. CMAQ chemical transport model and associated preprocessors 20
5.1. CMAQ core programs 48
5.2. BCON input and output files 50
5.3. CHEMMECH input and output files 66
5.4. ICON input and output files 70
5.5. JPROC input and output files 75
5.6. MCIP input and output files 78
5.7. PDM input and output files 84
5.8. PROCAN input and output files 89
6.1. Illustration of CMAQ boundary condition file 120
6.2. Graphical example of a CMAQ gridded boundary conditions file 121
6.3. Graphical example of a CMAQ gridded initial conditions file 122
7.1. Example sigma vertical structure in MM5. 139
7.2. Sample Arakawa-B staggering. 140
7.3. Sample domain and its nests. 142
8.1. Schematic of SMOKE approach to emissions processing 150

List of Tables

3.1. Software required for running CMAQ 29
3.2. Optional support software for CMAQ 30
4.1. Possible Time Step Structures in IO/API Files 36
4.2. Possible Data Type Structures in IO/API Files 37
4.3. Possible values for OPEN(3) FSTATUS 39
4.4. IOAPI data retrieval routines 40
4.5. CMAQ data manipulation utilities 42
5.1. BCON input files 51
5.2. BCON output files 55
5.3. CCTM input files 57
5.4. CCTM output files 65
5.5. CHEMMECH input files 67
5.6. CHEMMECH output files 69
5.7. ICON input files 71
5.8. ICON output files 74
5.9. JPROC input files 76
5.10. JPROC output files 77
5.11. MCIP input files 79
5.12. MCIP output files 82
5.13. PDM input files 85
5.14. PDM output files 88
5.15. PROCAN input files 89
5.16. Process Analysis Global Commands 91
5.17. Integrated Process Rate Output Command 91
5.18. IPR process codes 92
5.19. Integrated Reaction Rate Output Commands 93
6.1. CMAQ Input File Information 98
6.2. Coordinate system description segment 100
6.3. Coordinate system description segment 101
6.4. IC_PROFILE format description 103
6.5. BC_PROFILE format description 106
6.6. MECH_CONV_FILE format description 157
6.7. CSQY format description 112
6.8. Example CSQY data file 113
6.9. ET file format description 114
6.10. PROFILE file format description 116
6.11. 119
6.12. JTABLE file format description 124
9.1. Chemical Mechanisms 157

Chapter 1. Introduction

Table of Contents

1.1. Background and Goals 1
1.2. Overview of CMAQ System Components 3
1.3. Features of CMAQ for Application Users 6
1.4. Features of CMAQ for Air Quality Model Developers 6
1.5. New Features in Version 4.6 7
1.6. About this manual 9
1.7. References 9

The U.S. Environmental Protection Agencys Community Multiscale Air Quality (CMAQ) model is a three-dimensional Eulerian (i.e., gridded) atmospheric chemistry and transport modeling system that simulates ozone, acid deposition, visibility, and fine particulate matter throughout the troposphere. Designed as a one-atmosphere model, CMAQ can address the complex couplings among several air quality issues simultaneously across spatial scales ranging from local to hemispheric. The CMAQ source code is highly transparent and modular to facilitate extensibility through community development.

CMAQ is a third-generation air quality model that is designed for applications ranging from regulatory and policy analysis to understanding the complex interactions of atmospheric chemistry and physics. First-generation air quality models simulated air quality using simple chemistry at local scales, and Gaussian plume formulation was the basis for prediction. Second-generation models covered a broader range of scales (local, urban, regional) and pollutants, addressing each scale with a separate model that often focused on a single pollutant (e.g., ozone). Third-generation models, like CMAQ, treat multiple pollutants simultaneously up to continental or larger scales, often incorporating feedback between chemical and meteorological components. Future efforts toward fourth-generation systems will extend linkages and process feedback to include air, water, land, and biota to provide a more holistic approach to simulating transport and fate of chemicals and nutrients throughout an ecosystem.

1.1. Background and Goals

Air quality models integrate our understandings of the complex processes that affect the concentrations of pollutants in the atmosphere. Establishing the relationships among meteorology, chemical transformations, emissions, and removal processes in the context of atmospheric pollutants is the fundamental goal of an air quality model (Seinfeld and Pandis, 1998). In contrast to statistical air quality models that use historical trends in observed atmospheric conditions to predict air pollution, CMAQ uses coupled mathematical representations of actual chemical and physical process to simulate air quality. Based upon the underlying concept of preserving mass through a series of contiguous three-dimensional grid cells covering a fixed model grid, CMAQ belongs to the Eulerian class of mathematical models that calculate a mass balance within each grid cell by solving the transport across each cell boundary and chemical transformations within each cell during a given time period. As a framework for simulating the interactions of multiple complex atmospheric processes, CMAQ thus requires two major types of inputs: meteorological information and source emissions rates.

With weather conditions contributing the primary physical driving forces in the atmosphere (such as the changes in temperature, winds, cloud formation, and precipitation rates), representative gridded meteorology forms the basis of all 3-D air quality model simulations. The Fifth-Generation Pennsylvania State University/National Center for Atmospheric Research Mesoscale Model (MM5) (Grell et al., 1994) and the Weather Research and Forecasting (WRF) model Advanced Research WRF (WRF-ARW) (Skamrock et al., 2005) are two meteorological models that are compatible with CMAQ. The meteorology inputs dictate the following CMAQ configuration parameters:

- Horizontal grid coordinate and projection
- Horizontal grid resolution
- Maximum spatial coverage
- Vertical grid extent (model top)
- Temporal extent (start/end date/time, time step length)

For inputs on emissions, CMAQ uses an emissions model to estimate the magnitude, location, and temporal variability of pollution sources. Open-source models such as the Sparse Matrix Operator Kernel Emissions (SMOKE) model (CEP, 2006) and the Consolidated Community Emissions Processing Tool (CONCEPT) (http://www.conceptmodel.org/) are available for computing emissions inputs to CMAQ from annual, county-level emissions inventories. CMAQ emissions inputs must be on the same horizontal and vertical spatial scales and cover the same time period used in the air quality model simulation. The emissions inputs to CMAQ must also represent volatile organic compound (VOC) emissions using a chemical parameterization supported by CMAQ; currently supported photochemical mechanisms are the Carbon Bond IV (CB IV) mechanism (Gery et al., 1989; Dodge, 1989; Carter, 1996), the 2005 update to the Carbon Bond mechanism (CB05) (Yarwood et al., 2005), and the Statewide Air Pollution Research Center (SAPRC-99) mechanism (Carter, 1990, 2000). Additional details about the gas-phase chemistry in CMAQ are provided in Chapter 2 below and in Byun and Ching (1999). Those two sources also describe the primary aerosol emissions species that are supported by CMAQ. It is possible to add new emissions species to CMAQ that are not supported in the distributed version of the software using the chemical mechanism compiler (CHEMMECH) that is part of the CMAQ utility programs (see Chapter 5).

As a community model, the modular design of CMAQ facilitates customization and open-source development. Using the Input/Output Applications Programming Interface (I/O API) library (http://www.baronams.com/products/ioapi/AA.html) to control the internal and external data flows to the model, and the network Common Data Form (netCDF) library (http://www.unidata.ucar.edu/software/netcdf) to control the input and output file formats, CMAQ is based around a transparent and platform-independent code infrastructure that promotes extensibility by a broad community of developers. The modularity of CMAQ also leads to multiple science configuration options that model users can choose from when setting up new simulations. The trade-off for this flexibility is complexity in the model configuration; the model user is faced with hundreds of different configuration combinations when designing new simulations. To avoid confusion for new CMAQ users, this document provides guidance on a basic configuration to use for getting started with the model. For experienced air quality model users, the multiple configuration combinations available provide an unprecedented level of flexibility for optimizing model performance for different air quality model applications.

CMAQ is designed to meet the needs of the multiple groups contained within the air quality modeling community: research and regulatory modelers, algorithm and science module developers, air quality forecasters, and planners and policy makers. While each of these groups has distinct individual requirements of CMAQ, they also share a common need for an accurate, efficient, transparent, and economical tool that represents the state-of-the-science to simulate air pollution formation and transport. To address these individual and common needs, CMAQ has the following goals:

1. *Community Development* - Utilize a scientifically sound design that encourages innovations and enhancements by all members of the air quality modeling community
2. *Multiscale Modeling* - Provide adequate technical formulations to address air quality issues on multiple spatial scales, from urban to hemispheric
3. *One-atmosphere Design* - Provide robust and integrated science for modeling multiple, coupled air quality issues in a single simulation
4. *Modularity* - Maintain flexibility to add new or select from existing science modules to optimize model performance for specific applications
5. *Transparency* - Utilize programming practices that promote understanding of the model formulation at the source code level
6. *Computational Efficiency* - Provide scientifically acceptable results without compromising the speed at which the results are generated
7. *Open-Source Design* - Enable no-cost distribution and application by the modeling community

8. *State-of-the-Science* - Provide independent scientific peer review of all scientific processes within CMAQ on a recurring basis.

1.2. Overview of CMAQ System Components

CMAQ is a suite of FORTRAN-90 programs that work in concert to estimate ozone, particulate matter (PM), toxic compounds throughout the troposphere as well as acidic deposition. The five main CMAQ programs are

- The initial conditions preprocessor (ICON)
- The boundary conditions preprocessor (BCON)
- The clear-sky photolysis rate calculator (JPROC)
- The Meteorology-Chemistry Interface Processor (MCIP)
- The CMAQ Chemistry-Transport Model (CCTM)

Ancillary support programs distributed with CMAQ include

- The code builder/manager (M3BLD)
- The chemical mechanism compiler (CHEMMECH)
- The process analysis preprocessor (PROCAN)
- The plume dynamics model for plume-in-grid simulations (PDM)

This section describes the CMAQ system concept, followed by summaries describing each of the programs listed above.

All CMAQ source code except MCIP is distributed in a UNIX tar file as a Concurrent Versions System (CVS) (http://ximbiot.com/cvs/cvshome) source code management archive. CVS is an open source, network-transparent version control system for managing source code in a software development environment. CMAQ uses CVS to control access to the distributed source code by maintaining a protected version of the original source code and using copies of the code when building executables. CVS must be installed on the user's UNIX system before installing CMAQ.

When the distributed CMAQ tar file is unpacked, a CVS directory tree is installed on the user's machine that contains archived copies of the CMAQ code. The CMAQ program M3BLD controls the extraction of copies of CMAQ source code from CVS based on the configuration options specified by the user. After exporting the CMAQ source code from CVS, M3BLD then invokes a FORTRAN compiler to compile the CMAQ source code into FORTRAN object files and link them with the necessary precompiled libraries to create binary CMAQ executables. C and FORTRAN compilers must be installed on the user's UNIX system in order to create CMAQ executables.

Since the model infrastructure was designed to promote modularity,. CMAQ requires new executables for each suite of science configuration options for all programs except MCIP. There are too many combinations of the various chemical mechanisms, horizontal and vertical transport schemes, cloud routines, and chemistry solvers in the CMAQ science configuration to include in a single executable. The burden of recompiling CMAQ each time the science configuration changes is offset by the flexibility to add new science to the model or simply to switch between different configurations. This point about modularity is most pertinent to the CCTM, although there are configuration options that must be selected when compiling the other CMAQ programs.

In addition to compile-time configuration options with CMAQ, there are also execution-time configuration options. The horizontal domain configuration and the vertical coordinate are dynamic features in CMAQ that are independent of the executable. In other words, a user can employ a single executable for a simulation that uses any of the supported

map projections or grid definitions, without having to recompile the code. Distinctions between which CMAQ options must be selected at compilation versus at execution are described in Chapter 5.

As the chemistry-transport model component of CMAQ, the CCTM is the final program to be run in the CMAQ modeling sequence. The four other main programs prepare input data for the CCTM. Before describing each of the CMAQ programs (Section 1.2.4), we present a conceptual formulation of CMAQ and Eulerian air quality modeling to provide a framework for understanding the purposes and relationships of the various programs to each other and to the overall system.

Eulerian chemistry-transport models use coupled ordinary differential equations to solve the changes in concentration of pollutants throughout a three-dimensional grid that is fixed relative to a selected map projection. The changes in concentration in each grid cell are affected by the following processes:

- Emissions from sources
- Horizontal and vertical advection
- Horizontal and vertical diffusion
- Chemical transformations
- Deposition

Mathematically, these processes relate to the concentration change in each grid cell through the *continuity equation*, which is presented in simplified form below:

$$\delta C/\delta t = Adv + Diff + R_c + E_c - S_c$$

Adv = advection
Diff = diffusion
R_c = chemical transformation of species c
E_c = emissions of species c
S_c = loss processes for species c

In CMAQ, the advection and emissions terms are calculated based on input files generated by the meteorology and emissions models, respectively; the diffusion, chemical transformation, and loss process terms are calculated within the CCTM.

The Eulerian representation of the modeling domain is a series of contiguous grid cells that form a limited-area box on a subset of the globe, so the domain lateral boundary must define advection into the modeling grid. CMAQ currently accounts for advection into the domain only from the horizontal (i.e., lateral) boundaries, assuming there is no exchange through the top boundary of the domain. These spatial boundary conditions are estimated in CMAQ using the boundary conditions preprocessor, BCON. As a temporal boundary condition, the first time step of a model simulation is estimated in CMAQ using the initial conditions preprocessor, ICON. To model solar radiation, which provides the energy source for photolysis reactions, the program JPROC calculates clear-sky photolysis rates at various latitude bands and hours based on sun angles. Output from these CMAQ programs is used with output files from the emissions and meteorology models and other CMAQ preprocessors to form the required data for running the CCTM.

1.2.3. Summary Descriptions of the Major CMAQ Programs

The major CMAQ components and ancillary programs are the following:

- **Model Builder (M3BLD)** - The only CMAQ component written in C, M3BLD provides an interface to the CVS source code archive for exporting the source code, and to the FORTRAN compiler for building binary executables. As M3BLD is required to create all of the CMAQ executables except MCIP, it is the first program that needs to be compiled after installing the CMAQ source code on your system. In addition to creating executables, it also provides the option of generating a UNIX Makefile, which is particularly useful for porting the CMAQ code to

new operating systems, testing new code in a development environment, or troubleshooting problems with CMAQ compilation or execution.

- **Photolysis Rate Preprocessor (JPROC)** - JPROC calculates chemical-mechanism-specific clear-sky photolysis rates at fixed altitudes, hour angles, and latitude bands from tabulated absorption cross-section and quantum yield (CSQY) data. While CMAQ is distributed with CSQY data that support the default chemical mechanisms, updating or adding new CSQY data is straightforward. The only configuration option required for JPROC is the selection of the chemical mechanism to model. Output from JPROC is an ASCII lookup table of photolysis rates that the CCTM uses to calculate gas-phase chemical transformations and pollutant concentrations.
- **Initial Conditions Processor (ICON)** - ICON generates a gridded binary netCDF file of the chemical conditions in the modeling domain for the first hour of a simulation. It can generate these initial conditions from either an ASCII file of vertically resolved concentration profiles or from an existing CCTM output file. If the ASCII profiles do not have the same vertical structure as the CCTM configuration, ICON will interpolate the data to a vertical structure consistent with the CCTM's. Using an existing CCTM output file to generate initial conditions is applicable when extrapolating initial conditions from a coarse to a fine grid simulation, as may occur when setting up nested simulations. The configuration options for ICON include selecting the chemical mechanism to model, defining the horizontal and vertical grids, and choosing whether the initial conditions are generated from an ASCII profile or from an existing CCTM output file.
- **Boundary Conditions Processor (BCON)** - BCON generates a gridded binary netCDF file of the chemical conditions along the horizontal boundaries of the modeling domain. These boundary conditions can be either static or time-varying, and (as with ICON) can be generated from either an ASCII file of vertically resolved concentration profiles or from an existing CCTM output file. Also as with ICON, BCON will interpolate the data in the ASCII profiles to a vertical resolution that is consistent with the CCTM configuration. BCON differs from ICON in that it can generate time-varying (i.e., dynamic) boundary conditions. Dynamic boundary conditions are typically extracted from CCTM outputs from a coarse grid simulation for nested simulations or from a CTM simulation using a global-scale model. The file structure of the ASCII input profiles can also support the creation of dynamic boundary conditions, but generally these files are used for creating static data. The configuration options for BCON include selecting the chemical mechanism to model, defining the horizontal and vertical grids, and choosing whether the boundary conditions are generated from an ASCII profile or from an existing CCTM output file.
- **Meteorology-Chemistry Interface Processor (MCIP)** - MCIP uses MM5 or WRF output files to create netCDF-based input meteorology for the emissions model and the CCTM. MCIP prepares and diagnoses all meteorological fields that are required for the emissions model and the CCTM. In addition, MCIP is currently used to calculate the time-varying, species-dependent dry deposition velocities that are used in the CCTM. MCIP can be used to uniformly trim cells off the boundary of the domain defined by the meteorological model, or to window in on a subset of that domain. MCIP can also decrease the vertical resolution of the meteorological data by “layer collapsing”, although this option should be used with caution as it can degrade the quality of the data if used incorrectly. Configuration options for MCIP include the time periods over which to extract data from the meteorological model output files, horizontal and vertical grid information, and selections for either passing through certain MM5-calculated variables unaltered or recalculating these variables within MCIP.
- **CMAQ Chemistry-Transport Model (CCTM)** - The CCTM integrates the output from all of the preprocessing programs, including the emissions and meteorology models, to simulate continuous atmospheric chemical conditions. The concentrations of relevant species can be captured for output at a user-definable frequency (typically hourly). The CCTM output files (some of which are “optional”) are all binary netCDF files of gridded and temporally resolved air pollutant information, such as gas- and aerosol-phase species mixing ratios, hourly wet and dry deposition values, visibility metrics, and integral-averaged concentrations. The spatial and temporal extent of the CCTM output is dictated by the input meteorology. The science configuration is specific to each application of the model and can be adjusted to optimize model performance both computationally and in the numerical reproduction of observed air quality trends. Configuration options for the CCTM include the temporal coverage of the simulation, the chemical mechanism to model, the physics scheme to use for modeling pollutant transport, heterogeneous and aqueous chemistry options, plume-in-grid options, and diagnostic options (such as process analysis). The CCTM has the largest number of configuration options of all the CMAQ programs.
- **Plume Dynamics Model (PDM)** - The PDM is the plume-in-grid preprocessor to CMAQ. It generates plume dynamics, such as dimensions and location, from special emissions input files generated by an emissions model for plume-in-grid simulations (see Section 5.9).. The information generated by the PDM is supplied to a CCTM executable instrumented for plume-in-grid modeling that uses a Lagrangian reactive plume model for sub-grid-scale

representation of plumes from large point sources. The output from the PDM is a binary netCDF file of plume parameters for input to the CCTM.

- **Process Analysis Preprocessor (PROCAN)** - PROCAN generates FORTRAN INCLUDE files for building a version of the CCTM that can calculate integrated process rates and/or integrated reactive rates for diagnosing CMAQ simulations. It uses an input configuration file to select the processes analysis options, and outputs three INCLUDE files that are used to compile a version of the CCTM instrumented for process analysis.
- **Chemical Mechanism Compiler (CHEMMECH)** - This program creates mechanism INCLUDE files for CMAQ from a mechanism definition file. Chemical mechanisms are represented in CMAQ through a series of INCLUDE files that contain mechanistic and kinetic parameters that describe the mechanism. CHEMMECH creates these INCLUDE files from an ASCII mechanism definition file that represents the chemistry as sequential equations of reactants, products, and reaction rate information. This program is needed to modify reactions' stoichiometry or kinetics in the existing mechanisms, and to add new species that require the addition of new reactions.

More detailed discussions of the formulation of the above CMAQ programs are available in Chapter 2 below, in Byun and Ching (1999), and in Byun and Schere (2006).

1.3. Features of CMAQ for Application Users

CMAQ is designed to address the complex interactions among multiple air quality issues simultaneously. Using a one-atmosphere approach to air quality modeling by applying multiscale and multi-pollutant modeling techniques, CMAQ can provide independent but dynamically consistent predictions of several different pollutants at varying spatial scales. The modularity of the CMAQ design provides an unprecedented level of flexibility in air quality model configuration for optimizing model performance for different applications and spatial resolutions. In addition to the flexibility of the one-atmosphere approach, CMAQ provides the following features to users interested in applying the model for research or regulatory applications:

- Following the Models-3 paradigm, close interactions among the development communities for CMAQ and the meteorology and emissions models provide for a tight integration of the three main components of the air quality modeling system.
- Serial and multiprocessor execution options give the application user various options for optimizing the performance of CMAQ on different computer platforms.
- Community development expedites the expansion of CMAQ capabilities through the pursuit of multiple research agendas rather than from a single, centralized development group.
- A Comprehensive training program is available through the Community Modeling and Analysis System (CMAS) Center (http://www.cmascenter.org).
- The growing community of users connected through the CMAS Center is available to share data experiences and provide technical support.

1.4. Features of CMAQ for Air Quality Model Developers

Designed under a community modeling modeling paradigm, CMAQ is distributed as open source software engineered with a modular code design to facilitate decentralized development. Built around a layered I/O API and netCDF code framework, CMAQ provides an environment for testing new science algorithms, chemistry representations, and optimization techniques. CMAQ presents the following features to scientists interested in developing new algorithms or adding science to the model:

- Developed and distributed following open-source software conventions, CMAQ source code is easily accessible and free to obtain

- Designed for modularity, the CCTM uses standardized I/O routines to facilitate extensibility

- A growing community of CMAQ developers provides a forum for discussing development related topics

1.5. New Features in Version 4.6

CMAQ version 4.6 contains several new features and improvements from the previous release of the model. An important global update that affects ICON, BCON, and CCTM is the new dynamic layer configuration. Similar to the horizontal grid configuration that is defined at execution using a grid description file, the vertical layer configuration is now defined using a run-time environment variable that points to a 3-d input meteorology file that contains the applicable vertical structure. It is no longer necessary to recompile these programs when the vertical layer structure changes.

Additional, process-specific features are listed below.

- Aerosols
  - Updated ISORROPIA to v1.7 (includes correction in activity coefficients for temperatures other than 298 K
  - Revised to use T, RH-dependent “gamma” (heterogeneous N2O5 reaction probability) from Evans & Jacob (2005)
  - Upper limit set for the relative humidity input to ISORROPIA (95%)
  - Updated parameters written to the aerosol diagnostic file
- Chemistry
  - Carbon Bond (CB05) mechanism and associated EBI solver
  - Include the gas-phase reactions involving N2O5 and H2O
  - Removed obsolete mechanism combinations (e.g. gas+aerosols w/o aqueous)
- PBL modeling
  - New ACM2: combined non-local and local closure scheme
- Plume-in-Grid
  - Capability for AE4 mechanisms
- Mercury
  - Capability for CB05, AE4 mechanism
- Hazardous Air Pollutants
  - Increased number of gas-phase HAPs; added several toxic metals and diesel components in particulate material (adapted for CB05 and SAPRC99 chemical mechanisms, and AE4)
- Carbon Apportionment
  - Added capability for CB05 mechanism
- Sulfate Tracking
  - Added capability for : CB05, AE4 mechanisms ; and “cloud_acm” module

- CGRID restart file
  - Added a CGRID restart file (used to initialize next day's simulation; permits flexibility in the “CONC” file; can write subset of species and layers for the CONC file)
- Parallel I/O library
  - Various updates (new version required for CMAQv4.6 compilation)
- Other
  - Added a utility program to convert MCIP data into HYSPLIT format
- Meteorology (MCIPv3.2)
  - Modified I/O API header variable settings for WRF-ARW (WRF-EM) to reflect additional capability to have center of coarse domain located elsewhere than the center of the projection. (Note that applications of CMAQ with WRF-ARW input will no longer adhere to the “grid family” paradigm for the Models-3 I/O API, and a separate GRIDDESC file will be required for each grid.)
  - For non-PX runs only: Corrected error in bookkeeping “last wet time” in M3Dry dry deposition when Pleim-Xiu land-surface model is not run. Also now use ground temperature rather than 2-m air temperature to calculate ground-level relative humidity. Allow canopy to dry out gradually between 2 and 3 hours after “last wet time”.
  - Corrected checking and setting I/O API header variables for Mercator and polar stereographic projections.
  - Added restriction that Eta/Ferrier microphysics scheme in WRF-ARW (WRF-EM) is currently incompatible with CMAQ.
  - Added capability to fill land-use lookup table fields in MM5v3-formatted data that have been converted from MM5v2 format.
  - Corrected filling of vertical velocity array for MM5v2-formatted data.
  - Added land-water mask to output in GRIDCRO2D.
  - Corrected logic for processing MM5's TERRAIN file to account for expanded grid in coarse domain.
  - Updated condition for GRIDCRO3D output to reflect new WRF-ARW (WRF-EM) vertical coordinate descriptor in I/O API.
  - Optimized calculations of species-specific dry deposition constants.
  - Use land-water mask rather than dominant land use array to determine water points.
  - General code clean-up (removed unused variables, moved variables from global to local within subroutines, eliminated type mismatches, corrected/clarified comments).
  - Added ASCII file that includes calling tree for MCIP.
  - Updated frequently asked questions list.
  - Updated version stamp.

1.6. About this manual

This manual is an operational guidance document for users of the CMAQ modeling system, and is designed to support the installation, configuration, and execution of CMAQ on UNIX systems. Introductory and overview chapters present the CMAQ concepts, terminology, installation instructions, and guidance on running the tutorial simulation distributed with the model. Chapters on developing meteorology and emissions for CMAQ present the basic concepts and the availability of models that generate these critical CMAQ inputs, and describe how to configure these systems to produce input for CMAQ.

- Chapter 2 describes the science used in CMAQ.
- Chapter 3 describes computer system requirements and installation.
- Chapter 4 describes the Input/Output Applications Programming Interface (I/O API).
- Chapter 5 describes the set of CMAQ programs and libraries.
- Chapter 6 describes the CMAQ input files.
- Chapter 7 describes how to define the domain grids, layers, and chemical mechanisms.
- Chapter 8 describes setting up CMAQ for new simulations.
- Chapter 9 describes analysis options for CMAQ output.
- Chapter 10 describes CMAQ code management and development guidelines.
- Chapter 11 describes how to obtain support for CMAQ.

1.7. References

Byun, D., and J.K.S. Ching, 1999: Science Algorithms of the EPA Models-3 Community Multiscale Air Quality (CMAQ) Modeling System, United States Environmental Protection Agency Rep. EPA-600/R-99/030, 727 pp. [Available from Office of Research and Development, EPA, Washington, DC 20460

Byun, D., and K. L. Schere, 2006: Review of the governing equations, computational algorithms, and other components of the Models-3 Community Multiscale Air Quality (CMAQ) modeling system. *Appl. Mech. Rev*., **59**, 51-77.

Carter, W. P. L., 1996: Condensed atmospheric mechanisms for isoprene. *Atmos. Environ*., **30**, 4275-4290.

Carter, W. P. L., 1990: A detailed mechanism for the gas-phase atmospheric reactions of organic compounds. *Atmos. Environ*., **24A**, 481-518.

Carter, W. P. L., 2000: Implementation of the SAPRC-99 chemical mechanism into the Models-3 Framework. Report to the U.S. Environmental Protection Agency, 29 January 2000. [Available online at http://pah.cert.ucr.edu/~carter/reactdat.htm.]

CEP, 2006: SMOKE version 2.2 Users Manual. [Available online at http://cf.unc.edu/cep/empd/products/smoke/version2.2.]

Dodge, M. C., 1989: A comparison of photochemical oxidant mechanisms. *J. Geophys. Res*. **94**, 5121-5136.

Gery, M. W., G. Z. Whitten, J. P. Killus, and M. C. Dodge, 1989: A photochemical kinetics mechanism for urban and regional scale computer modeling. *J. Geophys. Res*., **94**, 12 925-12 956.

Grell, G. A., J, Dudhia, and D. R. Stauffer, 1994: A Description of the Fifth Generation Penn State/NCAR Mesoscale Model (MM5). NCAR Technical Note NCAR/TN-398+STR, 138 pp.

Seinfeld, J., and S. Pandis, 1998: Atmospheric Chemistry and Physics: From Air Pollution to Climate Change. John Wiley & Sons, New York, NY.

Skamarock, W. C., J. B. Klemp, J. Dudhia, D. O. Gill, D. M. Barker, W. Wang, and J. G. Powers, 2005: A Description of the Advanced Research WRF Version 2. NCAR Technical Note NCAR/TN-468+STR, 88 pp.

Yarwood, G., S. Rao, M. Yocke, and G. Whitten, 2005: Updates to the Carbon Bond chemical mechanism: CB05. Final Report to the U.S. EPA, RT-0400675. [Available online at www.camx.com.]

Chapter 2. Overview of the Science in the CMAQ Modeling System

Table of Contents

2.1. Features to Achieve the Goals of CMAQ 12
2.2. CMAQ Input Processors 14
2.3. CMAQ Chemical Transport Model Science Modules 19
2.4. The CMAQ User Interface 23
2.5. References 24

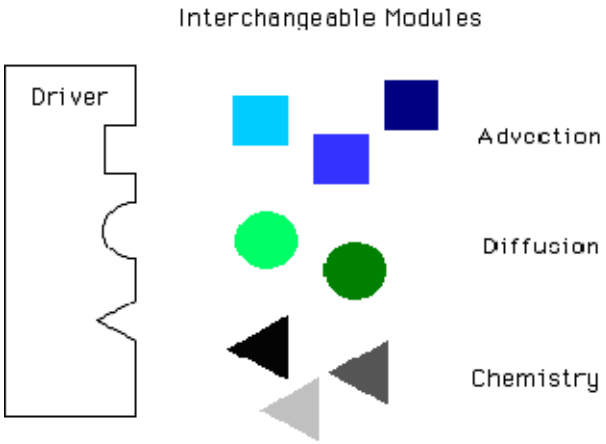
The National Ambient Air Quality Standards (NAAQS) were established by the U.S. Environmental Protection Agency (USEPA) under the authority of the Clean Air Act (U.S. EPA, 2004). These standards are designed to protect human health and the environment from high levels of criteria pollutants, such as ozone and particulate matter. The complex nature of air pollution scenarios requires control strategies to be effective for a variety of air pollutants, regions, and scales. In an effort to decrease ambient levels of criteria pollutants and to better understand the transport of chemicals, emission control strategies, and air quality, models have been approved by the USEPA for use at regional, state, and local scales within the United States. The models have been used to estimate the contribution of different control strategies to improve air quality and ensure cost effective results.

Historically models were designed and developed for specific air quality issues, such as acid deposition or ozone. Depending on the release characteristics, the pollutants in question, and the surrounding meteorological conditions at the time of pollutant release, modeling scenarios can range from a localized, short-term phenomenon to a long-term regional event. Because some emission sources contribute to the ambient levels of more than one pollutant and can affect an entire regional area on different time scales, an integrated modeling approach capable of handling multiple air pollutants and spatiotemporal scales was needed to isolate control strategies that improve overall air quality in a cost-effective manner. New air quality issues identified by the Clean Air Act Amendments of 1990 (such as visibility, fine and coarse particles, and indirect exposure to toxic pollutants) make an integrated modeling approach that can address multiple pollutants even more essential.

CMAQ is a multi-pollutant, multi-scale air quality modeling system that can simulate the transport and chemistry of ozone, particulate matter, toxic airborne pollutants, and acidic and nutrient pollutant species. CMAQ uses state-of-the-science techniques and has many new and important features that are not available in previous modeling systems. CMAQ is capable of modeling complex atmospheric processes affecting transformation, transport, and deposition of air pollutants using a system architecture that is designed for fast and efficient computing.

CMAQ allows users to easily construct models with different characteristics, such as different chemical mechanisms or alternative cloud treatments, in order to address a specific air quality issue (Figure 2-1). This modeling configuration will allow CMAQ to retain its state-of-the-science status over time with future implementations of new science modules as appropriate. At the same time, CMAQ can be employed for regulatory applications by using approved standard configurations of the modeling platform that represent the best available modeling technology at a given time. CMAQ has been developed to meet the needs of both the research and application communities.

Figure 2.1. CMAQ Modeling Framework



The CMAQ modeling system is also a useful tool for the model developer. It is unique in that its components are designed in a modular fashion with an interface that allow a model developer to create complex modeling situations and scenarios, or develop entirely new models using a standardized coding framework. Model developers can also perform sensitivity analyses on newly developed modules and perform comparisons with existing systems. Brief descriptions of the key features in CMAQ are discussed in this chapter. More detailed discussions on these features can be found in Byun and Ching (1999) and Byun and Schere (2006).

2.1. Features to Achieve the Goals of CMAQ

The CMAQ modeling system has a flexible modeling structure. Historically air quality model development resulted in separate air quality models that addressed specific issues. For instance, independent models were developed for ozone modeling and for acid deposition modeling. These models had little or no flexibility to be updated with advances in science or to accommodate new regulations. With this in mind, CMAQ was designed to have more adaptability and flexibility for different applications and for changing or improving methodology. The following subsections provide examples of how this is accomplished in CMAQ.

2.1.1. Multiple scales and multiple pollutants

CMAQ approaches air quality as a whole by including state-of-the-science capabilities for modeling multiple air quality issues, including tropospheric ozone, fine particles, toxins, acid deposition, and visibility degradation. The development of CMAQ includes the scientific expertise from each of these areas and combines the capabilities to enable a community modeling practice. CMAQ has multi-scale capabilities so that separate models are not needed for urban and regional scale air quality modeling. The target grid resolutions and domain sizes for CMAQ range spatially and temporally over several orders of magnitude. With the temporal flexibility of the model, simulations can be performed to evaluate longer term pollutant climatologies, as well as short-term transport from localized sources. By making CMAQ a modeling system that addresses multiple pollutants and different spatial scales, CMAQ has a “one atmosphere” perspective that combines the efforts of the scientific community. Developments and improvements to the CMAQ modeling system can continue as the state-of-the-science evolves. For example, linkages of CMAQ with exposure modeling, hydrological modeling, climate modeling, and multimedia modeling are currently under development.

To implement multi-scale capabilities in CMAQ, several different issues, such as scalable atmospheric dynamics and generalized coordinates that depend on the desired model resolution are addressed. Meteorological models may assume hydrostatic conditions for large regional scales, where the atmosphere is assumed to have a balance of vertical pressure and gravitational forces with no net vertical acceleration on larger scales. However, on smaller scales such as urban scales, the hydrostatic assumption cannot be made. A set of governing equations for compressible non-hydrostatic atmospheres is available to better resolve atmospheric dynamics at smaller scales, and they are more appropriate for

finer regional scale and urban scale meteorology. CMAQ's generalized coordinate system is used so that meteorological fields on different vertical coordinates can easily be accommodated and multiple scales can be simulated with the same CTM. The Jacobian used by the generalized coordinate system controls the necessary grid and coordinate transformations (cf., Byun 1999).

2.1.2. Modular flexibility

CMAQ's current coding structure is based on a modularity level that distinguishes its main driver, science modules, data estimation modules, and control/utility subroutines in the CCTM. The distinction remains at a division between the science models (including submodels for meteorology, emissions, chemistry-transport models), and analysis and visualization subsystems. In the CCTM, the process modules that affect the pollutant concentration fields are classified as follows:

Science Modules:

- Horizontal advection (*hadv*).
- Vertical advection (*vadv*).
- Mass conservation adjustments for advection processes (*adjc*).
- Horizontal diffusion (*hdiff*).
- Vertical diffusion (*vdiff*).
- Gas-phase chemical reaction solver (*chem*).
- Aqueous-phase reactions and cloud mixing (*cloud*).
- Aerosol dynamics and size distributions (*aero*).
- Plume chemistry effects (*ping*).

Control/Utility Modules:

- Model data flow and synchronizing of fractional time steps (*ctm*).
- Unit conversion (*gencoor*).
- Initialization (*init*).
- Process analysis (*pa*).

Data Estimation Modules:

- Aerosol deposition velocity estimation (*aero_depv*).
- Photolytic rate computation (*phot*).

This modularity makes it easier to modify or introduce a specific scientific process in the CCTM. For example, the *chem* module contains several options for different gas-phase chemistry solvers that can be used to optimize model performance. (More discussion will be provided about these modules later in this chapter.) Without the modular structure, such changes could entail coding modifications throughout the CCTM, thereby increasing the risk of human error. In that case, failing to make the correct modification at even a single location in the source code could lead to erroneous results or program execution failure. With the modularity featured in CMAQ, designing or modifying a model simulation is much more user-friendly, requiring less or no reprogramming.

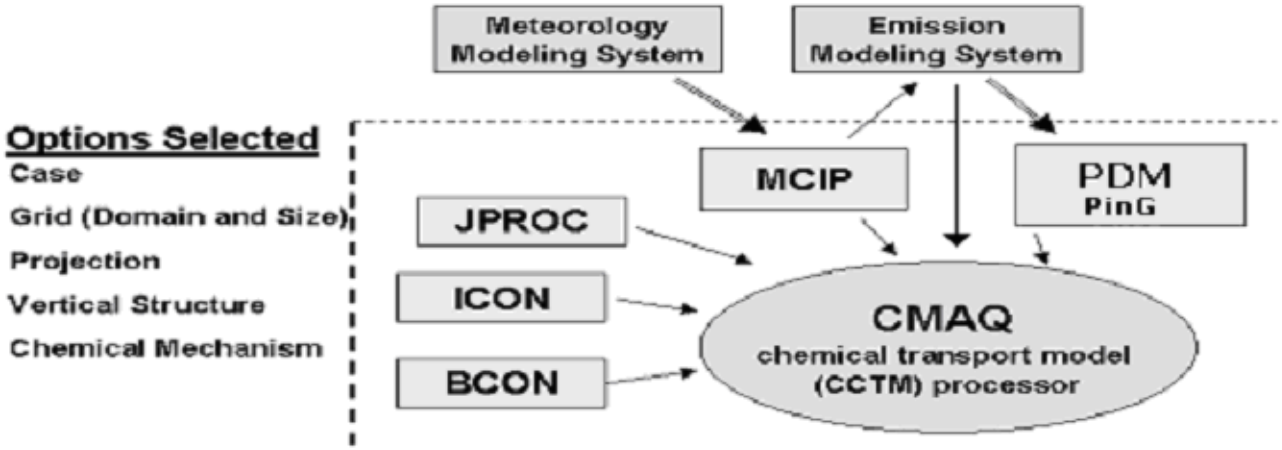
2.1.3. Quality control

CMAQ was designed to minimize the potential for model simulation error in several ways. For example, the generalized coordinate system helps to ensure that mass is conserved when transferring data from a non-hydrostatic meteorological model to the CMAQ modeling system. Otherwise, errors in air density and wind fields could induce large errors in the pollutant concentrations predicted by the air quality model. Also, the modularity of the scientific processes in CMAQ makes modifications and adaptations to the user's model more straightforward. The potential for error is minimized because the user is not required to change code or declare variables within the program modules. A common error in previous model formulations overlooked necessary program coding changes when making model modifications. Another feature of CMAQ that supports quality control is process analysis, which is discussed in Section 2.2.6.

2.2. CMAQ Input Processors

The CCTM uses data from other models and CMAQ subprograms as input for model simulations (Figure 2-2). Detailed information about the meteorological model in the context of preparing inputs for CMAQ can be found in Chapter 4. Detailed information about the emissions models for preparing CMAQ input can be found in Chapter 5.

Figure 2.2. Chemical Transport Model (CCTM) and preprocessors

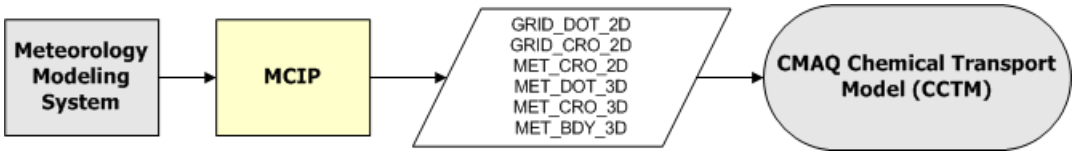


The input data for the CCTM are developed using the five processors shown in Figure 2-2. CMAQ uses the MCIP processor to prepare the meteorological fields for the CCTM. The ICON and BCON processors generate the initial and boundary conditions for a CCTM simulation. JPROC computes the photolysis rates that will be used when simulating photochemical reactions in the CCTM. The PDM generates plume information for emissions sources that use a subgrid Plume-In-Grid (PinG) treatment to characterize their emissions. Emissions for CMAQ must be prepared with a modeling system that generates emissions for direct input to the CCTM. Brief descriptions of the different CMAQ input processors are presented in this section.

2.2.1. MCIP: Meteorology Chemistry Interface Processor

The flow chart in Figure 2-3 illustrates the role of MCIP and its relationship to the CMAQ chemical transport model. In addition to the files shown in this Figure, GRIDDESC and GRID_CRO_3D provide necessary grid definition information for CMAQ execution.

Figure 2.3. Meteorology preprocessing for CMAQ with MCIP



Using output fields from the meteorology model, MCIP performs the following functions:

- Extracts meteorological model output for the CCTM horizontal grid domain. Important considerations here that CCTM uses a smaller computational domain than meteorological model, and the lateral boundaries from the meteorological model are generally not used by the CCTM.
- Processes all required meteorological fields for the CCTM and the emissions model.
- "Collapses" meteorological model fields, if coarser vertical resolution data are desired for the CCTM. MCIP uses mass-weighted averaging on higher vertical-resolution meteorological model output.
- If selected by the user, computes surface and planetary boundary layer (PBL) fields using output from the meteorological model.
- Computes dry-deposition velocities for important gaseous species using the surface and PBL parameters also generated by MCIP. Dry deposition is the deposition of pollutants from the air onto the surface of the earth. The rate of this removal is determined by various chemical, physical, and biological factors. In addition, dry deposition depends on the type of pollutant, nature and type of surface, and the amount of turbulence or mixing in the atmosphere. A measurement used in simulating the dry deposition of pollutants is the dry deposition velocity. The dry deposition velocity is analogous to the settling velocity of particles due to gravity, and it is useful in determining surface fluxes. MCIP can compute dry deposition using two methods:
  - The RADM dry deposition method (Wesely, 1989) calculates deposition velocities of 13 chemical species using friction velocities and aerodynamic resistances. Inputs required for this method include temperature, humidity, and horizontal wind component profiles.
  - The surface exchange aerodynamic method (Pleim et al., 2001) uses surface resistance, canopy resistance, and stomatal resistance to compute dry deposition velocities.
- Computes cloud top, cloud base, liquid water content, and cloud coverage for cumuliform clouds using simple convective schemes. The cloud parameters influence CCTM aqueous phase chemistry and cloud mixing as discussed in Section 2.2.5 (Walcek and Taylor, 1986; Chang et al., 1987). First, the cloud base is determined as the lifting condensation level computed from the highest saturated equivalent potential temperature below 700 mb. Then, the cloud top is computed by following a saturated adiabatic lapse rate from cloud base until it reaches a temperature five degrees cooler than the surrounding environment. Once the top and bottom of the cloud are determined, a vertical profile of the adiabatic liquid water mixing ratio can be constructed as the difference between the saturated mixing ratio at each level and the source level mixing ratio. By iteratively solving the equations governing the conservation of total water mass, energy conservation for cloud top mixing, along with the temperature profile, the cloud coverage fraction can be obtained.
- Outputs meteorological/geophysical files in the I/O API format, which is standard within the Models-3 framework.

2.2.2. ICON and BCON: The Initial and Boundary Conditions Processors

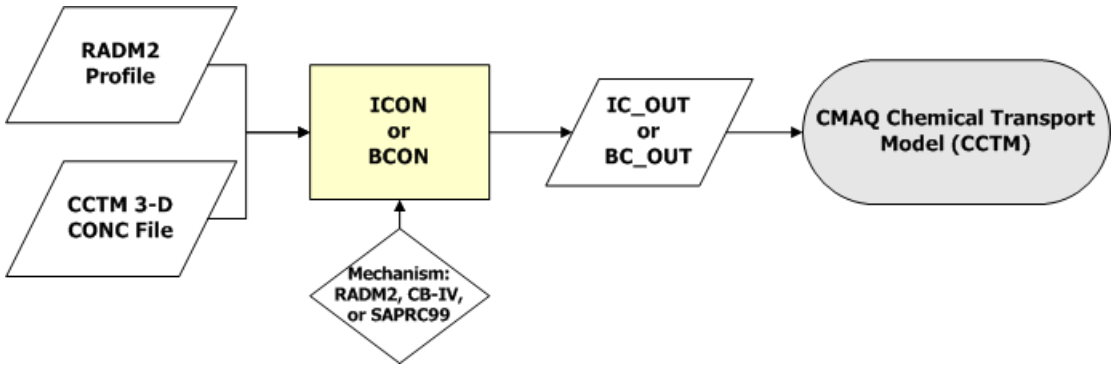
To perform air quality simulations, both initial and boundary conditions are required. Initial conditions (calculated in ICON) are needed to provide concentrations at the first time step of individual chemical species for the modeling domain.

Boundary conditions (calculated in BCON) are needed to provide concentrations of individual chemical species at the lateral boundaries of the modeling domain. ICON and BCON can generate these concentrations for multiple chemical species.

ICON and BCON require two inputs (Figure 2-4): a concentration file for the chemical species to be simulated and the chemical mechanism. The concentration file used in ICON and BCON can come from one of two sources as follows:

- Time independent set of vertical Regional Acid Deposition Model, Version 2 (RADM2) (Stockwell et al., 1990) concentration profiles. This approach is usually used when no other information about the initial and boundary concentrations is available.
- Existing CCTM 3-D concentration fields. Usually, this is used when performing a nested model simulation and modeling results from a previous CCTM simulation are available from a coarser grid resolution simulation. Existing CCTM concentration fields are also used when a CCTM simulation is extended in a separate run step. Unlike the profiles mentioned previously, these CCTM concentration files are spatially and temporally resolved.

Figure 2.4. Initial and boundary conditions preprocessing for CMAQ



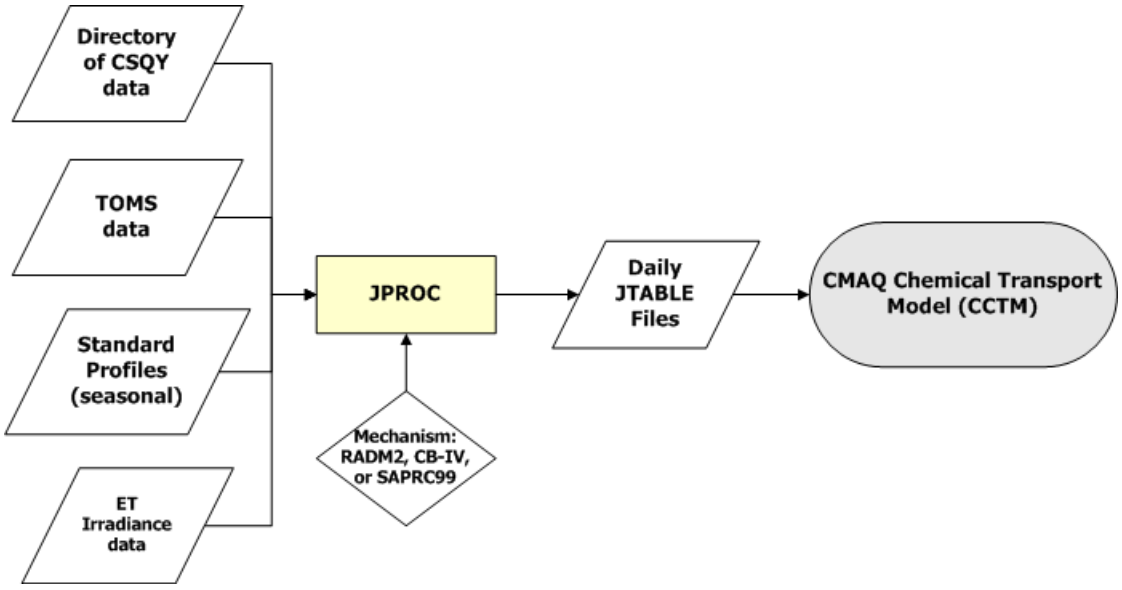
Both the vertical concentration profile fields and the CCTM concentration fields have specific chemical mechanisms associated with them, which are a function of how they were originally generated. Figure 2-4 assumes the RADM2 chemical mechanism was used to generate the default vertical profiles and is consistent with the profiles distributed with the CMAQ version 4.4 tutorial dataset. Existing CCTM 3-D concentration fields could have been generated using several different chemical mechanisms. The chemical mechanism used in the user's CCTM and CMAQ input processors must be consistent with the mechanism used to generate the ICON and BCON concentration fields, unless you use the option to convert to either Carbon Bond-IV (CB-IV) (Gery et al., 1989) or Statewide Air Pollution Research Center 1999 (SAPRC-99) (Carter, 2000) from RADM2. ICON and BCON are capable of converting from the RADM2 to either the CB-IV or SAPRC-99 chemical mechanisms, as discussed in Chapter 5.

ICON and BCON can linearly interpolate input concentration profiles from one horizontal or vertical coordinate system to the one needed for the model simulation if the input data are in the standard I/O API format. If the interpolation is between two different vertical coordinate systems, the mid-layer height of each vertical layer must also be available.

2.2.3. JPROC: Clear-Sky Photolysis Rate Calculator

For CMAQ, the photolysis rate model, JPROC, is used to generate clear sky photo-dissociation reaction rates. JPROC requires temperature profiles from the U.S. Standard Atmosphere (NOAA, 1976), a profile of the aerosol extinction coefficients, (Elterman et al., 1969) and can optionally use ozone column totals from the NASA Total Ozone Mapping Spectrometer (TOMS) satellite to produce the photolysis rates for CCTM (Figure 2-5).

Figure 2.5. Clear sky photolysis rate preprocessing for CMAQ

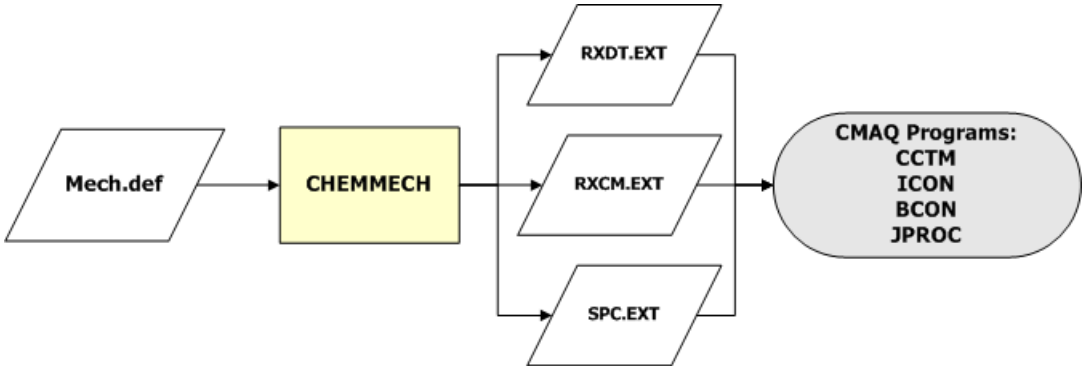


JPROC uses this information in a radiative transfer model to calculate the actinic flux (photons cm^{-2} min^{-1}) needed for calculating photolysis rates. Currently, JPROC calculates the actinic flux for clear-sky conditions (no clouds present), and CCTM then attenuates for cloudiness when clouds are present. JPROC computes the rate for each photolysis reaction at various latitudes, altitudes, and zenith angles. Within CCTM, the subroutine PHOT interpolates the data generated by JPROC to individual grid cells, and adjusts for the presence of clouds.

2.2.4. CHEMMECH: Chemical Mechanism Compiler

Gas-phase chemical mechanisms are implemented in CMAQ using FORTRAN include files. These files are in a machine-readable ASCII format and include all of the mechanism parameters including gas-phase species, reaction stoichiometry, and kinetics information. To invoke chemical mechanisms in CMAQ, these files are included in the compilation of the different CMAQ programs to generate mechanism-specific executables. CHEMMECH takes a mechanism definition file, often named mech.def, and generates the mechanism and species include files that define the chemistry parameters for the CMAQ programs. The file “mech.def” is a human-readable ASCII file that is easy to understand and modify. Figure 2.6 demonstrates the relationship between CHEMMECH and the rest of the CMAQ modeling system.

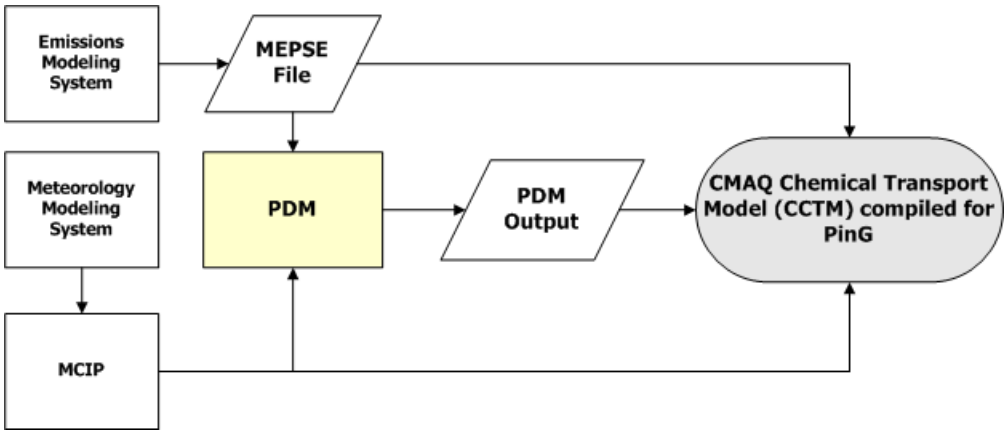
Figure 2.6. Invoking new/modified gas-phase chemical mechanisms in CMAQ



2.2.5. PDM: The Plume Dynamics Model for Plume-In-Grid

Plume in Grid (PinG) is a subgrid-scale treatment of the plumes from large point emissions sources. PinG is a technique to resolve the chemistry and dynamics of the plumes from selected point sources at a finer spatial resolution than the grid being modeled. Additional information about the PinG formulation in CMAQ is available in Byun and Ching (1999) and Byun and Schere (2006). The Plume Dynamics Model (PDM) is the PinG preprocessor in CMAQ that prepares special emissions files with information about sources selected for PinG treatment for input to the CCTM. The PDM simulates plume rise, horizontal and vertical plume growth, and transport of each plume section during the subgrid scale phase of the plume cycle. The PDM outputs an active plume file that is input to the CCTM along with the major elevated point source emissions (MEPSE) file from an emissions processing system to simulate selected point sources with the PinG treatment. Figure 2-7 is a schematic that relates PDM to other components in the CMAQ modeling system.
Figure 2.7

Figure 2.7. Plume in Grid treatment in the CMAQ modeling system

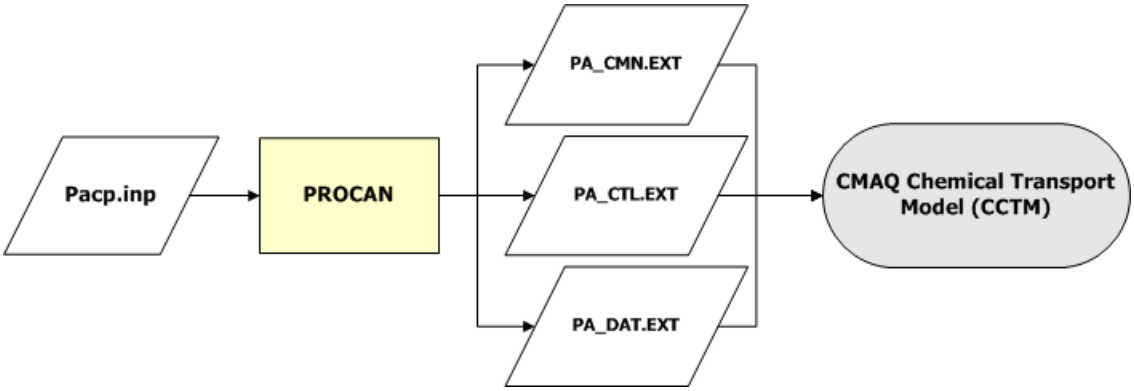


2.2.6. PROCAN: Process-Analysis Preprocessor

Process Analysis (PA) is an accounting system that tracks the quantitative effects of the individual chemical and physical processes that combine to produce the predicted hourly species concentrations within a CTM simulation. Typically Eulerian grid models, such as the CCTM, produce output concentration fields that are solutions of systems of partial differential equations for the time-rate of change in species concentrations due to a series of individual physical and chemical processes and then combining these results to obtain a cumulative hourly concentration. PA tracks the mass throughput of these individual processes and provides quantitative information about how each process impacted the predicted hourly species concentrations. PA is an optional configuration in CMAQ that is implemented by compiling the CCTM with include files that define PA configuration options. A PA-instrumented version of the CCTM outputs additional files that contain hourly concentrations by physical and/or chemical process for selected model species.

PROCAN is the PA preprocessor in CMAQ. PROCAN inputs a configuration file that is used to define the model species and processes to track using PA and outputs three include files that are used when compiling the CCTM. Figure 2-8 is schematic of PA in the CMAQ modeling system.

Figure 2.8. Process analysis implementation in the CMAQ modeling system



2.3. CMAQ Chemical Transport Model Science Modules

Figure 2-9 illustrates the CMAQ modeling system used to simulate the chemistry and transport of pollutants. Figure 2-9 also illustrates how CMAQ relates to other parts of the Models-3: the meteorological models, emissions model, and the analysis packages. For a model simulation, CMAQ needs input information including meteorological and emissions data. Using this information, the CMAQ Chemistry-Transport Model (CCTM) simulates each of the atmospheric processes that affect the transport, transformation, and removal of ozone, particulate matter, and other pollutants. CMAQ uses state-of-the-science techniques to simulate these processes, as described in the following subsections.

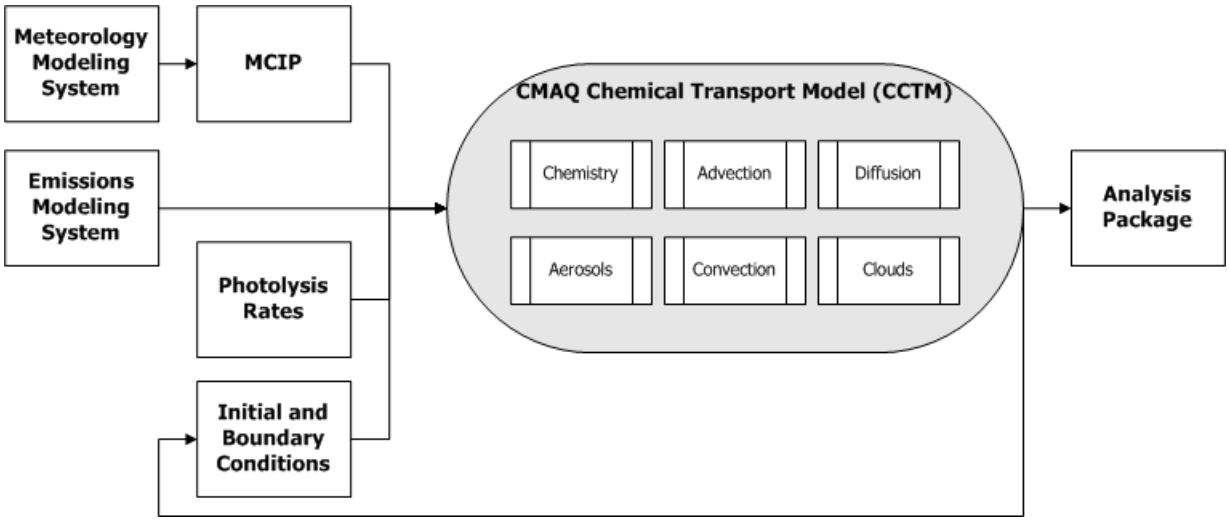
2.3.1. Gas-Phase Chemistry Solvers

Various modules for simulating tropospheric gas-phase chemistry within CMAQ have been developed, ranging from simple linear and nonlinear systems for engineering model prototypes to comprehensive chemistry representations for detailed chemical pathways related to atmospheric acid and oxidant formation. In CMAQ Version 4.6, gas-phase chemistry can be simulated with the Carbon Bond IV (CB-IV), the 2005 update to the Carbon Bond mechanism (CB05) (Yarwood et al., 2005), or the Statewide Air Pollution Research Center, Version 1999 (SAPRC99) photochemical mechanisms. Because of CCTM's modularity, users can modify the existing photochemical mechanisms or add new mechanisms. To compute time-varying species concentrations and their rate of formation or depletion (called chemical kinetics), equations governing chemical reaction kinetics and species conservation need to be solved for the entire species set. CCTM uses special numerical techniques called chemistry solvers that compute these concentrations and rates at each time step.

Different solution algorithms have been investigated for the chemical kinetics in terms of the optimal balance of accuracy, generalization, and computational efficiency that is required for this component of the atmospheric system. CCTM currently contains three options for solving gas-phase chemical transformations with the Rosenbrock (ROS3) (Sandu et al., 1997), Euler Backward Iterative (EBI) (Hertel et al., 1993), and Sparse Matrix Vectorized GEAR (SMVGEAR) (Jacobson and Turco, 1994) solvers. Version 4.6 includes the provision for mercury and other toxic compounds. This version also includes special routines to track sulfate and organic carbon species.

Different solution algorithms have been investigated for the chemical kinetics in terms of the optimal balance of accuracy, generalization, and computational efficiency that is required for this component of the atmospheric system. CCTM currently contains three options for solving gas-phase chemical transformations with the Rosenbrock (ROS3)[24], Euler Backward Iterative (EBI)[25] and Sparse Matrix Vectorized GEAR (SMVGEAR)[15] solvers.

Figure 2.9. CMAQ chemical transport model and associated preprocessors



2.3.2. Photolysis

Photolysis (or photo dissociation) of trace gases initiates most chemical reactions that take place in the atmosphere. Photolysis splits gas-phase chemical species with energy from sunlight. Photolysis is involved in the formation of smog, an air pollution problem that affects human, animal, and plant health. Simulating photochemical reactions accurately is therefore a key issue that strongly influences air quality model performance.

CCTM uses state-of-the-science techniques to simulate photolytic reactions in the Phot module. Photolysis reactions and their rates of reaction are driven by sunlight. Similar to kinetic reaction rates for non-photochemical reactions, the photolysis rate quantifies how much reactant is produced from a photolytic reaction in a given amount of time. The rate of photolysis is a function of the amount of solar radiation (called actinic flux), which varies based on the time of day, season, latitude, and terrestrial features. The amount of solar radiation is also affected by the amount of cloudiness and by aerosol absorption and scattering in the atmosphere. The photolysis rate also depends on species-specific molecular properties like the absorption cross-section (the effective molecular area of a particular species when absorbing solar radiation, which results in a shadow region behind the particle) and quantum yield (the number of molecules that dissociate for each light photon incident on the atmosphere). These molecular properties depend on the wavelength of the incident radiation and the temperature (and hence, on the available photon energy). Thus, estimating the photolysis rate is further complicated by these temperature and wavelength dependencies. As discussed in Section 2.2.3, the CMAQ modeling system includes an advanced photolysis model (JPROC) to calculate temporally varying photolysis rates for use in simulating photolysis in CCTM.

2.3.3. Diffusion and Advection

Pollutant transport includes both advection and sub-grid scale diffusion. Advection has to do with pollutant transport due to the mean wind fields, and diffusion involves sub-grid scale turbulent mixing of pollutants. If a plume is transported primarily by advection, then it may travel a long distance without much change in pollutant concentrations. On the other hand, if a plume is transported primarily by diffusion, then the pollutants will mix more quickly and nearer to the source, which will result in substantial changes to pollutant concentrations.

In CCTM, the advection process is divided into horizontal and vertical components. This distinction is possible because mean atmospheric motion is mostly horizontal. Often, the vertical motion is related to the interaction of dynamics and thermodynamics. The advection process relies on the mass conservation characteristics of the continuity equation. Data consistency is maintained for air quality simulations by using dynamically and thermodynamically consistent meteorology data from MCIP. When the meteorological data and the numerical advection algorithms are not exactly

mass consistent, one needs to solve a modified advection equation (Byun, 1999). A new mass continuity scheme, similar to that used in the air quality forecasting version of CMAQ, has been implemented for the 2005 release. This scheme is globally mass-conserving and uses the piecewise parabolic method (PPM) (Colella and Woodward, 1984) advection scheme for horizontal advection, deriving a vertical velocity component at each grid cell that satisfies the mass continuity equation using the driving meteorology model's air density. The mixing ratio correction step, used in previous CMAQ versions, is not needed with this method. Note that, the former advection scheme, with the same horizontal advection but also using PPM for the vertical velocity component, is still available, along with the mixing ratio correction step.

The horizontal advection module for CMAQ is the Piecewise-Parabolic Method (PPM) (Colella and Woodward, 1984). This algorithm is based on the finite volume subgrid definition of the advected scalar. In PPM, the subgrid distribution is described by a parabola in each grid interval. PPM is a monotonic and positive definite scheme. Positive definite schemes maintain the sign of input values, which means, in this case, that positive concentrations will remain positive and cannot become negative.

The vertical advection modules solve for the vertical advection with no mass-exchange boundary conditions at the bottom or top of the model. CMAQ also uses PPM as its vertical advection module. In CCTM, the PPM algorithm with a steepening procedure is implemented for vertical advection as the default because of the strong gradients in the tracer species that are observed in photochemical air quality conditions.

In CCTM, turbulent fluxes are expressed in terms of the mixing ratios and an air-density-weighted Jacobian to handle atmospheric diffusion processes with generalized coordinates. This approach is convenient for numerically solving the flux-form turbulence mixing because most flux-based closure algorithms use parameterizations of turbulent fluxes of conserving quantities, such as mass mixing ratios. In CMAQ version 4.5, Eddy diffusion is implemented for representing vertical diffusion. Eddy diffusivity is a local mixing scheme and is estimated using the same planetary boundary layer (PBL) similarity-based algorithm as in the Regional Acid Deposition Model, RADM, (Chang et al., 1987, 1990). In CCTM, the deposition process is simulated as a flux boundary condition that affects the concentration in the lowest layer. By treating the deposition process as the loss of mass due to the diffusion flux at the bottom of the model, one can relate the bottom boundary condition in the generalized coordinate system to that in the Cartesian coordinate system. Version 4.5 has an improved version of the minimum allowable vertical eddy diffusivity. The new version interpolates between urban and non-urban land cover allowing a larger minimum value for a larger fraction of urban land cover.

In CMAQ version 4.5 horizontal diffusion is implemented with a single eddy diffusion algorithm that is based on local wind deformation and is scaled to the grid cell size. The horizontal eddy diffuseness is assumed to be uniform but dependent on the grid size of the model. The diffusivity is larger for a higher resolution run where the numerical diffusion due to the advection process is smaller.

2.3.4. Particulate Matter

CCTM uses a modal approach to simulate PM2.5 (particulate matter of diameter equal to or less than 2.5 microns), coarse particulate matter (particulate matter of diameter greater than 2.5 microns and equal to or less than 10 microns), and PM10 (particulate matter of size equal to or less than 10 microns). PM2.5 is further divided into the Aitken and accumulation modes. Coarse particulate matter is represented currently by fugitive dust and a generic anthropogenic species. PM10 is the sum of the PM2.5 and coarse particulate matter.

Particulate matter in the atmosphere can either be primary or secondary. Primary particulate matter is emitted directly into the atmosphere from natural or anthropogenic emissions. Secondary particulate matter is formed in the atmosphere either from precursors, as a result of chemical reactions, or from condensation or deposition onto primary particles that are already present in the atmosphere. CCTM can predict fine particulate speciated concentrations of sulfate, nitrate, ammonium, organics, and aerosol water, and it includes methods for simulating nucleation, dry deposition, and cloud processes. The 3rd generation CMAQ aerosol module (aero3) takes chemical species concentrations and reactivity rates from the chemistry solvers and primary particulate concentrations from the emissions processor to compute fine and coarse particulate concentrations.

Invoking the AERO3 module will influence the sulfate, ammonia, and nitric acid concentrations predicted by the gas phase chemistry module. The sulfate is partitioned between the vapor (sulfuric acid) and particle phases. The greater part of the sulfate is put into the aerosols with a very small residual amount remaining in the vapor phase. Nitric acid and ammonia are equilibrated with the aerosol species.

The aerosol module in CMAQ version 4.5 is called AERO4. The difference between AERO4 and AERO3 is that AERO4 includes consideration of sea salt aerosols. Emissions of sea salt from the open ocean are calculated as a function of wind speed and relative humidity, following the parameterizations of Gong (2003) and Zhang et al. (27), respectively. These emissions are speciated into sodium, chloride, and sulfate, and are distributed by size to the accumulation and coarse modes by fitting the emission parameterization to a bimodal distribution. Note that sea-salt emissions are calculated within the CMAQ model and do not require special pre-processing in SMOKE. Equilibrium between the accumulation mode (which now contains sodium and chloride) and the gas phase (which now contains hydrochloric acid) is calculated within the ISORROPIA thermodynamic module. As in previous CMAQ releases, mass transfer is not simulated between the coarse mode and the gas phase. These sea-salt processes can be activated by selecting the AERO4 module and deactivated by using AERO3. For easier comparison of CMAQ output with measurement, AERO4 now provides three new variables (PM25AT, PM25AC, and PM25CO) that are the fractional amounts of the Aitken, accumulation, and coarse modes, respectively, that are composed of particles less than 2.5 m in aerodynamic diameter (Jiang et al., 2006)

Unlike gases, the deposition velocity for particles must be calculated from the aerosol size distribution, as well as meteorological and land-use information. The aero_depv2 module calculates the size distribution from the mass and number concentration for each of the three modes and calculates the dry deposition velocity. The dry-deposition algorithm has been modified to include an impaction term in the coarse and accumulation modes. Finally, two diagnostic tools, for tracking the sulfate budget and sources of elemental and primary organic carbon have been added.

CCTM's AERO module is also useful for evaluating visibility. CCTM integrates Mie scattering (a generalized particulate light-scattering mechanism that follows from the laws of electro-magnetism applied to particulate matter) over the entire range of particle sizes to obtain a single visibility value for each model grid cell at each time step. A more detailed description of the particulate matter simulation techniques used in CCTM can be found in Binkowski and Shankar (1995), and Byun and Schere (2006).

2.3.5. Clouds and Aqueous-phase Chemistry

Clouds are an important component of air quality modeling and play a key role in aqueous chemical reactions, vertical mixing of pollutants, and removal of pollutants by wet deposition. Clouds also indirectly affect pollutant concentrations by altering the solar radiation, which, in turn, affects photochemical pollutants, such as ozone, and the flux of biogenic emissions. The Cloud module in CMAQ performs several functions related to cloud physics and chemistry. Three types of clouds are modeled by CMAQ: sub-grid convective precipitating clouds, sub-grid non-precipitating clouds, and grid-resolved clouds. The subgrid cloud scheme in the CCTM was derived from the diagnostic cloud model in RADM, Version 2.6 (Dennis et al., 1993; Walcek and Taylor, 1986; Chang et al, 1987; Chang et al., 1990). Grid resolved clouds are diagnosed by MM5 with no additional cloud dynamics considered in CMAQ. The aqueous chemistry model in CMAQ evolved from the original RADM model (Chang et al., 1987, and Walcek and Taylor, 1986). The cloud module in the CCTM vertically redistributes pollutants for the sub-grid clouds, calculates in-cloud and precipitation scavenging, performs aqueous chemistry, and accumulates wet deposition amounts. Version 4.5.1 contains an important improvement in the convective cloud mixing algorithm that corrects a tendency to over-predict excessive transport from upper layers in the cloud to sub-cloud layers.

2.3.6. Plume-in-Grid Modeling (PinG)

When individual sources are added directly into a grid model, it is assumed that the emissions are instantaneously mixed into the grid where the source is located. In reality, stack gases are emitted in conical plumes that disperse and expand gradually as a function of stability, wind speed, and distance downwind of the stack. Thus, the assumption that individual sources are instantaneously mixed can cause over-dilution of the emissions into large grid cells. Also, when

advection dominates as the transport mechanism for the emissions, the source's plume can be mixed too close to the source, when it should be mixed into air downwind of the source location.

Improvements have yielded a more realistic treatment of the subgrid-scale physical and chemical processes impacting individual point-source plumes. CCTM's Plume Dynamics Model (PDM) treats plume rise, vertical/horizontal growth due to turbulence and sheer processes, and plume position. A Lagrangian Reactive Plume Module (LRPM) accounts for the relevant dynamic and chemical processes of the subgrid plumes and incorporates the plume material into the intercepted grid cells, when certain physical size or chemical criteria are met. PinG is designed for the larger scale grid-size simulations (36 km and 12 km), and it is not invoked for the smaller scale 4-km simulations. At fine resolutions, the emissions are released directly into the grid cells, and subgrid level plume resolution is not needed. It is assumed that instantaneous mixing occurs at this fine resolution.

2.3.7. Process Analysis

CCTM also includes a process analysis (pa) module. Process analysis is a technique for quantifying the contributions of various physical and chemical processes, and the changes in the predicted concentrations of a pollutant. Process analysis is useful in the quality control of a model because it separates out the influence of individual processes on the pollutant concentration. With this information, compensating or unresolvable errors in the model or input data can be identified even if they are not reflected in the total change in concentration. For example, if an error in the emission input data caused negative concentration values, it may not be recognized in the predicted concentration because of a large addition in the predicted concentration resulting from chemical processes.

In addition to its role in the quality control of air quality modeling, process analysis is a very strong analysis tool. As a tool for model development, process analysis can help evaluate the effect of modifications on a model or process module. As a tool for regulatory decision making, process analysis can help determine if a decision to control a specific type of emission would produce the desired results, or if another control would be more effective.

2.4. The CMAQ User Interface

The distributed CMAQ user interface consists of a series of C-shell scripts for building and running the different CMAQ programs on Linux operating systems. These scripts can be adapted to work with any Unix shell scripting language (e.g. Bash, Bourne, etc.) and function primarily to set a series of environment variables required by M3BLD or the CMAQ program executables. Manipulating the CMAQ scripts requires the use of a Unix text editor such as *vi* (http://www.vim.org/) or *nedit* (http://www.nedit.org/). Alternatives to Unix shell scripts exist in prototype CMAQ graphical users interfaces (GUI) with the Multimedia Integrated Modeling System (MIMS) and the CMAQ GUI, both available through the CMAS Center (http://www.cmascenter.org).

There are three basic User Interfaces for CMAQ; the CMAQ C-shell scripts (the distributed user interface), the CMAQ Graphical User Interface (GUI), and the CMAQ Unified Archive. The C-shell scripts and the Unified Archive are both text-based UNIX interfaces in which CMAQ is configured and compiled by editing the scripts. These scripts can be adapted to work with any Unix shell scripting language (e.g. Bash, Bourne, etc.) and function primarily to set a series of environment variables required by M3BLD or the CMAQ program executables. Manipulating the CMAQ scripts requires the use of a Unix text editor such as vi (http://www.vim.org/) or nedit (http://www.nedit.org/). Alternatives to Unix shell scripts exist in prototype CMAQ graphical users interfaces (GUI) with the Multimedia Integrated Modeling System (MIMS) and the CMAQ GUI, both available through the CMAS Center (http://www.cmascenter.org). The CMAQ GUI is a java based program that is designed to download, compile, configure, and run CMAQ all through its interface. All three interfaces are available for download at the CMAS center website, www.cmascenter.org. The CMAS Center currently fully supports CMAQ on Linux systems using the Portland Group compiler; there is limited support for compiling and running CMAQ on other UNIX computing systems.

2.4.1. C-shell Scripts

The CMAQ C-shell scripts are available in separate tar files on the CMAS Center website. The MODELS file contains all of the source code for CMAQ and the SCRIPTS file contains all of scripts for building and executing the individual programs. Each of CMAQ's programs has a separate build script and a separate run script. The build script is used to compile the program and the run script is used to set the required environment variables and run the program. You can edit the build script and the run script using a text editor such as the vi editor. There are certain options that need to be set at compilation and some that can be set before running a simulation. All changes must be done by editing the individual scripts for each program. There are also previous releases of CMAQ available for download at the CMAS Center website. Details about using the scripts to build and run CMAQ are described in Chapter 3, *CMAQ System Requirements and Installation* [27].

2.4.2. CMAQ GUI

The CMAQ GUI is a windows-like interface that runs on Linux systems and helps the user download, compile, and run CMAQ. It is designed to be user-friendly and uses both a normal and an advanced view to better accommodate different skill levels. The GUI is divided into six basic parts and begins by downloading CMAQ and the associated libraries into user-defined directories. It then helps the user to select the compiler flags and compile the libraries. The next step is to compile the various programs that make up CMAQ. The GUI will notify the user of any error that occurs while downloading or compiling. The different options and environment variables can be set by simply choosing them from drop down menus in the interface. After compiling all programs the user can then run CMAQ from the GUI. The user also have the option to either run the CMAQ program in batch mode, running all the programs sequentially or you can run the programs individually. The latter option allows the user to make small changes in an individual program and not have to run the entire simulation over again.

2.4.3. Unified Archive

The Unified Archive is a consolidated scripting interface for CMAQ. The Unified Archive interface refines the compiling of the CMAQ programs and the necessary libraries (except for the netCDF library) into a single Makeall script. The Makeall script is edited with a text editor similar to the original C-shell scripts. The Makeall script can be configured to either compile all of the CMAQ programs and libraries with one execution or to only compile selected system components. There is also a master run script that consolidates the running of all the programs into a single script. The Unified Archive is designed to be simpler and faster than editing each individual build and run script as in the original C-shell interface. The Unified Archive contains all past releases of CMAQ and allows the user to optionally choose which version of CMAQ to compile.

2.5. References

Binkowski, F.S., and U. Shankar, 1995: The Regional Particulate Model: Part I. Model description and preliminary results. *J. Geophys. Res.*, **100**, 26 191–26 209.

Binkowski, F. S., and S. J. Roselle, 2003: Models-3 Community Multiscale Air Quality (CMAQ) model aerosol component. 1. Model description. *J. Geophys. Res.*, **108**, 4183, doi:10.1029/2001JD001409.

Bott, A., 1989: A positive definite advection scheme obtained by nonlinear renormalization of the advective fluxes. *Mon. Wea. Rev.*, **117**: 1006–1015.

Byun, D. W., 1999: Dynamically consistent formulations in meteorological and air quality

models for Multiscale atmospheric studies. Part I: Governing equations in a generalized coordinate system. *J. Atmos. Sci.*, **56**, 3789–3807.

Byun, D. W., and J. K. S. Ching, 1999: Science Algorithms of the EPA Models-3 Community Multiscale Air Quality (CMAQ) Modeling System. U. S. Environmental Protection Agency Rep. EPA-600/R-99/030, 727 pp. [Available from Office of Research and Development, EPA, Washington, DC 20460.]

Byun, D., and K. L. Schere, 2006: Review of the governing equations, computational algorithms, and other components of the Models-3 Community Multscale Air Quality (CMAQ) modeling system. *Appl. Mech. Rev.*, **59**, 51–77.

Carter, W. P. L., 2000: *Documentation of the SAPRC-99 chemical mechanism for VOC reactivity assessment.* Final Report to California Air Resources Board, Contract No. 92-329 and 95-308.

Chang, J. S., R. A. Brost, I. S. A. Isaksen, S. Madronich, P. Middleton, W. R. Stockwell, and C. J. Walcek, 1987: A Three-Dimensional Acid Deposition Model: Physical Concepts and Formulation. *J. Geophys. Res.*, **92**, 14 681-14 700.

Chang, J. S., P. B. Middleton, W. R. Stockwell, C. J. Walcek, J. E. Pleim, H. H. Lansford, F. S. Binkowski, S. Madronich, N. L. Seaman, D. R. Stauffer, D. Byun, J. N. McHenry, P. J. Samson, and H. Hass, 1990: The regional acid deposition model and engineering model, *Acidic Deposition: State of Science and Technology*, Report 4, National Acid Precipitation Assessment Program.

Coats, C., 2005: The EDSS/Models-3 IO/API. [Available online at http://www.baronams.com/products/ioapi.]

Colella, P., and P. L. Woodward, 1984: The piecewise parabolic method (PPM) for gas-dynamical simulations. *J. Comput. Phys.*, **54**, 174–201.

Dennis, R. L, J. N. McHenry, W. R. Barchet, F. S. Binkowski, and D. W. Byun, 1993: Correcting RADM's sulfate underprediction: Discovery and correction of model errors and testing the corrections through comparisons against field data. *Atmos. Environ.*, **26A**, 975–997.

Elterman, L., R. Wexler, and D. T. Chang, 1969: Features of tropospheric and stratospheric dust. *Appl. Optics*, **8**, 893–903.

Gery, M. W., G. Z. Whitten, J. P. Killus, and M. C. Dodge, 1989: A photochemical kinetics mechanism for urban and regional scale computer modeling. *J. Geophys. Res.*, **94**, 12 925–12 956.

Gong, S. L., 2003: A parameterization of sea-salt aerosol source function for sub- and super-micron particles. *Global Biogeochem. Cycles*, **17**, 1097, doi:10.1029/2003GB002079.

Hertel O., R. Berkowicz, J. Christensen, and O. Hov, 1993: Test of two numerical schemes for use in atmospheric transport-chemistry models. *Atmos. Environ.*, **27A**, 2591–2611

Jacobson, M., and R. P. Turco, 1994: SMVGEAR: A sparse-matrix, vectorized Gear code for atmospheric models. *Atmos. Environ.*, **28**, 2991–3003.

Jiang, W., S. Smyth, E. Giroux, H. Roth, and D. Yin, 2006: Differences between CMAQ fine

mode particle and PM2.5 concentrations and their impact on model performance evaluation in the lower Fraser valley. *Atmos. Environ.*, **40**, 4973–4985.

Pleim, J. E., and J. S. Chang, 1992: A non-local closure model in the convective boundary layer. *Atmos. Environ.*, **26A**, 965–981.

Pleim, J. E., A. Xiu, P. L. Finkelstein, and T. L. Otte, 2001: A coupled land-surface and dry deposition model and comparison to field measurements of surface heat, moisture, and ozone fluxes. *Water Air Soil Pollut. Focus*, **1**, 243–252.

Sandu, A., J. G. Verwer, J. G., Blom, E. J. Spee, G. R. Carmichael, and F. A. Potra, 1997: Benchmarking stiff ODE solvers for atmospheric chemistry problems. II: Rosenbrock solvers. *Atmos. Environ.*, **31**, 3459–3472.

Smolarkiewicz, P. K., 1983: A simple positive definite advection scheme with small implicit diffusion. *Mon. Wea. Rev.* **111**, 479–486.

Smyth, S. C., W. Jiang, D. Yin, H. Roth, and E. Giroux , 2006: Evaluation of CMAQ O_3 and $PM_{2.5}$ performance using Pacific measurement data. *Atmos. Environ.*, **40**, 2735–2749.

Stockwell, W. R., P. Middleton, J. S. Chang, and X. Tang, 1990: The second generation regional acid deposition model chemical mechanism for regional air quality modeling. *J. Geophys. Res.*, **95**, 16 343–16 367.

U.S. EPA, 2004: National Ambient Air Quality Standards (NAAQS). [Available online at http://www.epa.gov/air/criteria.html.]

U.S. Standard Atmosphere, 1976: National Oceanic and Atmospheric Administration, U.S. Government Printing Office, Washington, DC, NOAA-S/T76-1562.

Unidata, 2005: NetCDF. [Available online at http://my.unidata.ucar.edu/content/software/netcdf/index.html.]

Walcek, C. J., and Taylor, G. R., 1986: A Theoretical Method for Computing Vertical Distributions of Acidity and Sulfate Production within Cumulus Clouds. *J. Atmos. Sci.*, **43**, 339–355.

Wesely, M. L., 1989: Parameterization of surface resistances to gaseous dry deposition in regional-scale numerical models. *Atmos. Environ.*, **23**, 1293–1304.

Yarwood, G., S. Rao, M. Yocke, and G. Whitten, 2005: Updates to the Carbon Bond chemical mechanism: CB05. Final Report to the U.S. EPA, RT-0400675. [Available online at www.camx.com.]

Zhang, K. M., E. M. Knipping, A. S. Wexler, P. V. Bhave, and G. S. Tonnesen, 2005: Size distribution of sea-salt emissions as a function of relative humidity. *Atmos. Environ.*, **39**, 3373–3379.

Chapter 3. CMAQ System Requirements and Installation

Table of Contents

3.1. System Recommendations 27
3.2. Installing CMAQ Source Code 30
3.3. Compiling CMAQ for the Tutorial Simulation 32
3.4. Running the CMAQ Tutorial Simulation 34
3.5. Benchmarking CMAQ 34

This chapter provides recommended hardware configurations and software requirements for installing and running CMAQ. The hardware configurations in particular are subject to change with each new release of CMAQ and with the development of new computing technologies. The installation instructions in this chapter guide the user through the obtaining the CMAQ source code and unpacking it on your system. Brief instructions for running the CMAQ tutorial case and benchmarking the model are also addressed. Here, the term “benchmarking” refers to the process of verifying that a model has installed correctly on a new computer. Configuring CMAQ for new applications is covered in Chapter 7.

3.1. System Recommendations

All of the CMAQ programs (with the exception of M3BLD) are written in FORTRAN and are optimized for use on computers running a version of the UNIX operating system. Most personal computers (PC’s) running the Linux operating system are sufficiently powerful to handle basic CMAQ applications. However, to use CMAQ in a production environment where multiple iterations of the model will be executed for different spatial domains and/or emissions control strategies, either a cluster of multi-processor PC’s on a high-end network or an expandable rack-mounted UNIX server is recommended. In light of the dynamic nature of the computer hardware world, the specifications listed in the section are current recommendations, not requirements. While there are minimum levels of processing power and disk capacity needed for running CMAQ, there is no single system that CMAQ is optimized to run on. The flexibility of the modeling system enables the users to optimize CMAQ for most current hardware configurations.

CMAQ is distributed and supported for executing on Red Hat Linux operating systems with the Portland Group Fortran compiler. Ports have been made to other brands of Linux such as Mandrake and Suse, to other Fortran compilers, such as Absoft and Intel, and to other hardware systems such as Sun, Silicon Graphics, and Macintosh. Information about these ports and up-to-date hardware recommendations are able through the CMAS Center website (http://www.cmas-center.org/). The hardware recommendations provided below are the minimum level required to run CMAQ, these certainly can be expanded to more powerful and larger systems with benefits to the model performance. The software recommendations are closer to requirements as all of the necessary source code libraries and utilities needed for running CMAQ are listed.

3.1.1. Hardware

The minimum hardware requirements to run CMAQ are:

- PC with a single 1.0 GHz processor with a Red Hat Linux OS
- 1 GB RAM
- 40 GB hard drive storage

Below are two examples of optimal hardware configurations for running CMAQ on multiple processors in a production environment:

Optimal CMAQ Hardware Solution # 1

- 4 dual-CPU 2.8 GHz Xeon IBM BladeCenter rack-mounted nodes with Red Hat Enterprise Linux OS
- 2.5 GB RAM per node
- Gigabit Ethernet network
- 1.5 TB hard drive storage
- Super DLT 110 GB tapes for system back-ups
- UPS

Optimal CMAQ Hardware Solution #2

- 8 dual-CPU 2.5 GHz AMD Athlon MP 2000+ PC’s with Red Hat Fedora Core 1
- 2.0 GB RAM per PC
- Gigabit Ethernet network
- 80 GB system storage
- 10 TB IDE-SCSI RAID 5 array
- Super DLT 110 GB tapes for system back-ups
- UPS

3.1.2. Software

To run CMAQ, the programs listed in Table 3-1 must be installed on the chosen system; this list includes the programs distributed with CMAQ. Table 3-2 lists additional utility software that is not required for running CMAQ, but is useful for model diagnostics and evaluation.

Table 3.1. Software required for running CMAQ

Software	Description	Source
CMAQ Programs		
M3BLD	Models-3 program builder for source code management and code compilation	Contained in the standard CMAQ distribution available at http://www.cmas-center.org/
JPROC	Photolysis rate processor	
ICON	Initial conditions preprocessor	
BCON	Boundary conditions preprocessor	
MCIP	Meteorology-Chemistry Interface Processor for converting MM5 output to CMAQ format	
CCTM	CMAQ Chemical Transport Model	
PDM	Plume Dynamics Model for calculating plume characteristics for Plume in Grid modeling	
CHEMMECH	Chemical mechanism compiler for modifying or adding reactions to the CMAQ chemistry	
PROCAN	Process analysis processor for setting up CMAQ to generate integrated reaction rates or integrated process rates	
Compilers		
PGF90	Portland Group, or Intel FORTRAN 90 compiler	http://www.pgroup.com/
GCC	Gnu C compiler	http://gcc.gnu.org/
Code libraries		
STENEX	CMAQ stencil exchange library for parallel job management	Contained in the standard CMAQ distribution available at http://www.cmas-center.org
PARIO	CMAQ parallel input output management library	
netCDF	Network Common Data Form library for controlling CMAQ file formats	http://my.unidata.ucar.edu/content/software/netcdf/index.html
IOAPI	Input/Output Application Programmers Interface for controlling internal and external communications	http://www.baronams.com/products/ioapi/
Source code management		
CVS	Concurrent Versions System for managing the distributed archive of the CMAQ source code	http://ximbiot.com/cvs/cvshome/

Table 3.2. Optional support software for CMAQ

Software	Description	Source
Evaluation and visualization tools		
PAVE	Package for Analysis and Visualization of Environmental Data for graphical analysis of netCDF gridded data	http://www.cmascenter.org
IDV	Integrated Data Viewer for 3-D graphical analysys of netCDF gridded data	http://www.unidata.ucar.edu/software/idv/
IO API Tools	Postprocessing tools for manipulating data in the IO API/netCDF format	http://www.baronams.com/products/ioapi/
netCDF Tools	Postprocessing tools for manipulating data in the netCDF format	http://my.unidata.ucar.edu/content/software/netcdf/index.html
Source code diagnostics		
PGDBG	Portland Group FORTRAN 90 debugger	http://www.pgroup.com/
PGPROF	Portland Group FORTRAN 90 code profiler	http://www.pgroup.com/

3.2. Installing CMAQ Source Code

After installing CVS, the IOAPI and netCDF libraries, and Fortran and C compilers on the hardware system, download the CMAQ source code, scripts, and benchmark data files from the CMAS Center website (http://www.cmascenter.org). After registering to download CMAQ on the CMAS Center Software Clearinghouse users are redirected to a page that contains links to download UNIX tar files of the CMAQ code, scripts, and benchmark data along with various documents describing the installation and execution processes.

The following files and archives comprise the CMAQ distribution:

- README - text file describing the CMAQ installation
- CHEM_SOLVER_NOTES - text file containing comments on chemical solvers available in CMAQ and their relative computational performance and numerical accuracy
- CVS_NETCDF - text file explaining the CVS configuration management system and the netCDF data system and how to set them up
- IOAPI - text file explaining how to get the I/O API system and set up the libraries
- SMOKE - text file describing the SMOKE emissions system and how to set it up (not required for the tutorial simulation)
- PARALLEL_NOTES - text file containing comments related to running the CMAQ CCTM in Linux MPICH clusters
- PROCESS_ANALYSIS_NOTES - text file containing comments related to running the CMAQ CCTM with process analysis
- PING_PDM_NOTES - text file containing comments related to running the CMAQ CCTM with plume-in-grid treatment
- TUTORIAL_PROCEDURE - text file describing how to work through the tutorial
- RELEASE_NOTES - text file containing a list of the changes since the last release
- MODELS.tar.gz - gzipped tar file (~2.3 Mb) containing model, tools and libraries source code CVS archives
- M3DATA.tar.gz - gzipped tar file (~106 Mb) containing the required datasets not produced by this package
- SCRIPTS.tar.gz - gzipped tar file (~17 Kb) containing C-Shell scripts to build and execute the CMAQ models
- M3DATA_REF.tar.gz - gzipped tar file (~315 Mb) containing reference data to compare with datasets produced by the tutorial on a Linux workstation

The default CMAQ installation is configured for a Red Hat Linux system, and the Portland Group FORTRAN compiler.

1. Set the environment variable M3HOME to alias the directory in which you want to install CMAQ, e.g.:

2. Set environment variables for the M3DATA, M3LIB, and M3MODEL directory paths needed by CMAQ:

   ```
   setenv M3DATA $M3HOME/data
   setenv M3LIB $M3HOME/lib
   setenv M3MODEL $M3HOME/models
   ```

3. Create the M3LIB directory

4. Download the CMAQ tar files and unpack them in the M3HOME directory using the following commands:

   ```
   tar xvzf SCRIPTS.tar.gz
   tar xvzf MODELS.tar.gz
   tar xvzf M3DATA.tar.gz
   tar xvzf M3DATA_REF.tar.gz
   ```

5. The CMAQ scripts assume that the netCDF and IOAPI are already installed on your system. You can either copy the precompiled libraries or create symbolic links to the library *.a files into the appropriate locations in the M3LIB directory. For example to copy a precompiled version of the netCDF library use a variation of the following commands:

   ```
   mkdir -p $M3LIB/netCDF/Linux
   cp /home/user/netcdf-3.6.1/lib/libnetcdf.a $M3LIB/netCDF/Linux
   ```

   or to create a symbolic link:

   ```
   mkdir -p $M3LIB/netCDF/Linux
   cd $M3LIB/netCDF/Linux
   ln -s /home/user/netcdf-3.6.1/lib/libnetcdf.a libnetcdf.a
   ```

   Similarly for the IOAPI, either copy a precompiled version of the library to the $M3LIB/ioapi/Linux2_x86pg directory or create a symbolic link to the library *.a file.

   Compiler flag consistency between the netCDF and IOAPI are critical for building library files compatible with CMAQ. For Linux systems with the Portland Group Fortran compilers use the PGI Fortran and Gnu C compilers for building both the IOAPI and netCDF.

   Before compiling the netCDF set the following environment variables:

   ```
   CC = gcc
   CPPFLAGS = '-DNDEBUG -DpgiFortran`
   CFLAGS = -O
   FC = pgf90
   FFLAGS = `-O -w`
   CXX = g++
   ```

   To compile the IO API, set the BIN environment variable to Linux2_x86pg_gcc_mp before running the Unix “make” command:

```
BIN = Linux2_x86pg_gcc_mp
```

After unpacking all of the CMAQ tar files in the M3HOME directory, setting the appropriate environment variables, and installing the IOAPI and netCDF libraries in the CMAQ $M3LIB directory your system is ready to build the CMAQ executables. Before proceeding check that CVS is available on your system by typing the following command:

If the message “cvs: Command not found.” is returned you must install CVS on your system before you can continue with the CMAQ installation.

3.2.3. Notes on the CMAQ directory structure

The CMAQ installation includes a tutorial dataset for benchmarking the model on your system. Unpacking the different tar files of the distribution in the M3HOME directory installs the CMAQ source code, scripts, and tutorial data files in a directory structure recognized by the default run and build scripts. The M3HOME directory is the base location of the CMAQ installation for a specific application. Under M3HOME, the *scripts* directory contains the build and run scripts, the *models* directory contains the CVS archive of model source code, the *data* directory contains the input and output data for the model, and the *lib* directory contains the compiled binary library files required to build the CMAQ executables. The CMAQ scripts use the following environment variables to alias the locations of these different directories:

```
M3LIB   =      M3HOME/lib
M3DATA  =      M3HOME/data
M3MODEL =      M3HOME/models
```

Using environment variables to specify the CMAQ directories generalizes the CMAQ scripts by only requiring that the user specify the location of the M3HOME directory on their system and then setting the rest of the CMAQ directories relative to M3HOME. While this directory structure works well for the tutorial case and most CMAQ applications, other configurations are possible.

3.3. Compiling CMAQ for the Tutorial Simulation

For all CMAQ programs, other than MCIP, the program M3BLD is used to compile the source code into executables. The first step in the compilation of CMAQ is to compile M3BLD. Then compile the program libraries Stenex and Pario before moving on to the compilation of the rest of the CMAQ programs. *For all of the CMAQ programs and libraries, the directory paths for the Fortran and C compilers in the build scripts will have to be changed to reflect the correct locations on the users system.* For parallel compilations, some of the programs also require directory paths to MPICH include files. Program-specific compilation instructions are provided below. The compilation instructions in this section are for building executables for simulating the tutorial data sets distributed with CMAQ. Other than the paths to the compilers, the build scripts used in this section will not need to be changed to create working executables. Additional information about the configuration options for the different CMAQ programs are provided in Chapter 2 and 6.

1. Use the following commands to compile M3BLD:

   ```
   cd $M3HOME/scripts/build
   bldit.m3bld
   ```

2. Next create the stencil exchange libraries for serial and parallel processing. Check that the MPICH include file directory path is correct for the parallel version of the library on line 51 of the file bldit.se.pgf. Use the following commands to compile the Stenex libraries:

```
cd $M3HOME/scripts/stenex
bldit.se.pgf
bldit.se_noop.pgf
```

3. For parallel CCTM operation create the parallel I/O library:

```
cd $M3HOME/scripts/pario
bldit.pario.pgf
```

Now create the model executables for JPROC, ICON, BCON, and the CCTM.

4. For the benchmark case, confirm that JPROC is configured to produce photolysis rates for the mechanism labeled "cb05_ae4_aq". Use the following command to compile JPROC:

```
cd $M3HOME/scripts/jproc
bldit.jproc.pgf
```

5. ICON and BCON can be configured for different chemical mechanisms and for different kinds of input data. The configuration options for ICON and BCON are discussed in detail in Chapters 2 and 6. Use the following commands to compile ICON and BCON:

```
cd $M3HOME/scripts/icon
bldit.icon.pgf
cd $M3HOME/scripts/bcon
bldit.bcon.pgf
```

6. MCIP is compiled using a FORTRAN Makefile instead of M3BLD. To create the MCIP executable, set the compiler, compiler flags, and netCDF and I/O API library paths in the Makefile distributed with MCIP. Use the following commands to compile MCIP:

```
cd $M3HOME/scripts/mcip/BLD
make
```

7. CCTM has multiple configuration options that can be changed to optimize model performance for different applications. In addition to selecting the chemical mechanism to model, the user can also select from several different science modules. The science configuration options for CCTM are discussed in detail in Chapters 2 and 6. The CCTM build script is configured by default to run in multiprocessor mode for the benchmark simulation. CMAQ uses the MPICH message passing interface (MPI) to manage communication between processors in a clustered multiprocessor computing environment. Before compiling the CCTM for parallel execution, the user must specify the location of the MPICH directory on your system on line 156 of the CCTM build script. For single-processor systems or first-time users of the model, configure the build script to create a single-processor executable by commenting out the line that activates the variable "ParOpt" on line 52 of the build script. Use the following commands to compile CCTM:

```
cd $M3HOME/scripts/cctm
bldit.cctm.pgf
```

Although they are not used for the benchmark simulation, PDM and PROCAN can also be compiled using M3BLD.

3.4. Running the CMAQ Tutorial Simulation

After successfully compiling the various CMAQ programs, use the distributed run scripts to generate the CCTM input files and then to run the CCTM for the CMAQ benchmark case. The CCTM must be run last in the simulation sequence; there are no other dependencies between the other CMAQ programs (except that MCIP must provide necessary information to ICON and BCON), so they can be run in any order to create input data for the CCTM. If all of the programs compiled and the CMAQ environment variables are set correctly, then for all programs other than the CCTM the benchmark simulation will execute without any modifications to the run scripts. The CCTM is configured to run in multiprocessor (parallel) mode and requires setting the number of processors to allocate to the simulation and the location of the MPI initialization command on your system.

3.4.1. Notes on running the CCTM in parallel mode

Configuring the CCTM run script for parallel processing requires selecting the number of processors to use for the simulation by setting the NPROCS environment variable, and choosing the domain decomposition configuration by setting the variable NPCOL_NPROW. The number of processors must be equal to the product of the two values selected for NPCOL_NPROW. For example, if your system has six processors available to run CMAQ, set NPROCS to 6 and NPCOL_NPROW equal to “3 2”. For single-processor computing set NPROCS to 1 and NPCOL_NPROW to “1 1”. Most clustered multiprocessor systems require a command to start the MPI run-time environment. The CCTM run script is configured by default to use the *mpirun* command. Consult your system administrator to find out how to invoke MPI when running multiprocessor applications.

3.5. Benchmarking CMAQ

CMAQ is distributed with a dataset that can be used to benchmark the CMAQ installation. In the CMAQ distribution, output data from CMAQ is bundled with the input data (including emissions and meteorology) that can be used to reproduce the reference results.

After completing the CMAQ benchmark case, the CCTM output files can be compared with the reference datasets provided in the distribution. If the benchmark case is run on a system similar to that used to create the reference data (Red Hat Linux with the Portland Group FORTRAN compiler), the results should differ by no more than 0.01% for every model species. Changing the optimization of the compiler or compiling on other operating systems with different compilers can lead to larger differences between the benchmark results and the reference datasets. The CCTM benchmark targets for operating systems/compilers other than the Red Hat Linux/Portland Group combination are differences less than 1% for every model species.

Chapter 4. Input/Output Applications Programming Interface

Table of Contents

4.1. Files, Logical Names and Physical Names: 35
4.2. IO/API Data Structure and Data File Types 36
4.3. Opening/Creating Data Files in IO/API 38
4.4. Reading Data Files in IO/API 40
4.5. Writing Data Files in IO/API 40
4.6. CMAQ-Related I/O API Utilities 41
4.7. References 46

The Models-3 Input/Output Applications Programming Interface (I/O API) is an environmental software development library which provides an easy-to-learn, easy-to-use interface with the data involved in CMAQ applications (Coats, 2005). The files are self-describing Network Common Data Form (NetCDF) format files in which the file headers have all the dimensioning and descriptive information needed to define the resident data. According to Unidata (2005), NetCDF is an interface for array-oriented data access and a freely-distributed collection of software libraries for several languages that provide implementations of the interface.

The CMAQ data files all have selective direct access which means the modeling system can be more efficient by only reading the necessary parts of the data files. Additionally, I/O API files are portable across computing platforms. This means that the same file can be read, for example, on a Sun workstation, a Red Hat Linux workstation, and on a Mac OS X. A further benefit of the I/O API is that an expansive set of data manipulation utilities and statistical analysis programs are available to help evaluate and post-process the binary CMAQ input/output data files.

The I/O API system is essentially transparent to those CMAQ users employing a preconfigured application of the model. For users planning to modify the code or implement updated modules for research purposes, a few key elements of the I/O API should be understood and are discussed below. This section covers only the barest of necessities in terms of a CMAQ user's interaction with I/O API. For more detailed information about developing new modules for CMAQ using the I/O API code libraries, please refer to Chapter 9.

4.1. Files, Logical Names and Physical Names:

The I/O API stores and retrieves data using files and virtual files, which have (optionally) multiple time steps of multiple layers of multiple variables. Files are formatted internally so that they are machine and network-independent. This behavior is unlike FORTRAN files, whose internal formats are platform-specific, so that the files do not transfer using the File Transfer Protocol (FTP) or Network File System (NFS)-mount very well. Each I/O API file has an internal description, consisting of the file type, the grid and coordinate descriptions, and a set of descriptions for the file variables (i.e., names, unit specifications, and text descriptions). According to the I/O API format, files and variables are referred to by names, layers by number (from one to the greatest number of layers in the file), and dates and times are stored as integers, using the coding formats: YYYYDAY (commonly called "JDATE") and HHMMSS (commonly called "JTMIE") as defined in equations 4.1a and 4.1b.

$$\text{YYYYDAY} = (1000 * \text{Year}) + \text{Julian Day} \quad (4.1a)$$

$$\text{HHMMSS} = (10000 * \text{Hour}) + (100 * \text{Minute}) + \text{Seconds} \quad (4.1b)$$

Rather than forcing the programmer and program-user to deal with hard-coded file names or hard-coded unit numbers, the I/O API utilizes the concept of logical file names. The modelers can define the logical names as properties of a program and then at run-time the logical names can be linked to the actual file name using environment variables. For programming purposes, the only limitations are that file names cannot contain blank spaces and must be at most 16 characters long. When a modeler runs a program that uses the I/O API, environment variables must be used to set the values for the program's logical names. This will be discussed further in terms of CMAQ applications in Chapter 5. The remainder of this section will discuss some of the rudimentary details of programming in an environment using I/O API data files.

4.2. IO/API Data Structure and Data File Types

Each CMAQ data file has an internal file description that contains the file type, the file start date and time, the file time step, the grid and coordinate descriptions, and a set of descriptions for the set of variables contained within the file (i.e., names, units specifications, and text descriptions). Some of the items in a file description, such as the dates and times for file creation and update, and the name of the program which created the file, are maintained automatically by the system. The remainder of the descriptive information must be provided at the time of file creation.

All files manipulated by the I/O API may have multiple variables and multiple layers. Each file has a time step structure that is shared by all of its variables, as well. There are three kinds of time-step structure supported, as listed in Table 4-1. Within a file, all the variables are data arrays with the same dimensions, number of layers and the same data structure type, although possibly different basic types (e.g., gridded and boundary variables can not be mixed within the same file, but real and integer variables can). The data type structures that are supported are listed in Table 4-2. GRDDED3 and BNDARY3 are the most prevalent file types in a CMAQ simulation.

Table 4.1. Possible Time Step Structures in IO/API Files

File Type	Description
Time-independent	The file's time-step attribute is set to zero. Routines which deal with time-independent files ignore the date and time arguments
Time-stepped	The file has a starting date, a starting time, and a positive time step. Read and write requests must be for some positive integer multiple of the time step from the starting date and time.
Circular-buffer	This type of file keeps only two “records”, the “even”" part and the “odd” part (useful, for example, for "restart" files where only the last data written in the file are used). The file's description has a starting date, a starting time, and a negative time step (set to the negative of the actual time step). Read and write requests must be for some positive integer multiple of the time step from the starting date and time, and they must reflect a specific time step that is in the file.

Table 4.2. Possible Data Type Structures in IO/API Files

File Type	Magic Number	Data Type	Description
CUSTOM3	-1	Custom	User-dimensioned array of REAL*4s that these system reads/writes reliably
DCTNRY3	0	Dictionary	Data type stores and retrieves parts of an FDESC.EXT file description
GRDDED3	1	Gridded	Dimension as REAL*4 ARRAY (NCOLS, NROWS, NLAYS, NVARS)
BNDARY3	2	Boundary	Dimension as REAL*4 ARRAY (SIZE, NLAYS, NVARS)
IDDATA3	3	ID-reference	Used to store lists of data like pollution monitoring observations
PROFIL3	4	Vertical profile	Used to store lists of vertical data, like rawinsonde observations
GRNEST3	5	Nested grid	Preliminary and experimental implementation for storing multiple grids, which need not in fact have any particular relationship with each other beyond using the same coordinate system
SMATRX3	6	Sparse matrix	Sparse matrix data, which uses a "skyline-transpose" representation for sparse matrices, such as those found in SMOKE
KFEVNT3	-3	Cloud event	KF-Cloud files use the same file description data structures (from FDESC3.EXT) and defining parameters (from PARMS3.EXT); the usual I/O API DESC3() call may be used to retrieve file descriptions from the headers. KF-Cloud file, on the other hand, have their own specialized opening/creation, lookup/indexing, input, and output operations

File Type	Magic Number	Data Type	Description
TSRIES3	7	Hydrology Time Series	A hydrology time series file behaves much like a degenerate gridded file, except that the numbers of rows and columns are usually 1, and that there are additional file attributes found in the include file ATDSC3.EXT
PTRFLY3	8	Pointer-flyer	A pointer-flyer observation file behaves much like a degenerate gridded file with NCOLS3D and NROWS3D set to 1, and certain mandatory variables and variable-naming conventions to be used by analysis and visualization software

4.3. Opening/Creating Data Files in IO/API

The I/O API function OPEN3 was created to open both new and existing files. OPEN3 is a FORTRAN logical function that returns TRUE when it succeeds and FALSE when it fails.

LOGICAL FUNCTION OPEN3(FNAME, FSTATUS, PGNAME)

where: CHARACTER*(*) FNAME: file name for query

INTEGER FSTATUS: see possible values in Table 4.3

CHARACTER*(*) PGNAME: name of calling program

This function maintains considerable audit trail information in the file header automatically, as well as automates various logging activities. The arguments to OPEN3 are the name of the file, an integer FSTATUS indicating the type of open operation, and the caller's name for logging and audit-trail purposes. OPEN3 can be called many times for the same file. FSTATUS values are defined for CMAQ in PARMS3.EXT and are also listed in Table 4.3.

Table 4.3. Possible values for OPEN(3) FSTATUS

FSTATUS	Value	Description
FSREAD3	1	for READ-ONLY access to an existing file
FSRDWR3	2	for READ/WRITE/UPDATE access to an existing file
FSNEW3	3	for READ/WRITE access to create a new file (file must not yet exist)
FSUNKN3	4	for READ/WRITE/UPDATE access to a file whose existence is unknown
FSCREA3	5	for CREATE/TRUN-CATE/READ/WRITE access to files

In the last three cases, “new” “unknown” and “create/truncate,” the code developer may fill in the file description from the INCLUDE file FDESC3.EXT to define the structure for the file, and then call OPEN3. If the file doesn't exist in either of these cases, OPEN3 will use the information to create a new file according to your specifications, and open it for read/write access. In the "unknown" case, if the file already exists, OPEN3 will perform a consistency check between your supplied file description and the description found in the file's own header, and will return TRUE (and leave the file open) only if the two are consistent.

An example of OPEN3 function is listed below (from the CMAQ INITSCEN subroutine). This program segment checks for the existence of a CCTM concentration (CTM_CONC_1) file, which if found will be open read-write-update. If the CCTM CONC file is not found, a warning message will be generated.

```
IF ( .NOT. OPEN3( CTM_CONC_1, FSRDWR3, PNAME ) ) THEN

      MSG = 'Could not open ' // CTM_CONC_1 // ' file for update - '

  &       // 'try to open new'

      CALL M3MESG( MSG )

END IF
```

To get a file description (i.e., I/O API file type, dimensions, start date, start time, etc.), can be optained bye using DESC3. When DESC3 is called, the complete file description is placed in the standard file description data structures in FDESC3.EXT . Note that the file must have been opened prior to calling DESC3. A typical call might look like this:

```
IF ( .NOT. DESC3( ' myfile' ) ) THEN

  ... error message

ELSE

  ... DESC3 commons now contain the file description

END IF
```

4.4. Reading Data Files in IO/API

There are four routines with varying kinds of selectivity used to read or otherwise retrieve data from files: READ3 , XTRACT3, INTERP3, and DDTVAR3. All four are logical functions that return TRUE when they succeed, FALSE when they fail. The descriptions of the routines are listed in Table 4.4.

Table 4.4. IOAPI data retrieval routines

Routine	Description
READ3	reads one or all variables and layers from a file for a particular date and time.
XTRACT3	reads a windowed subgrid for one or all variables from a file for a particular
INTERP3	interpolates the requested variable from the requested file to the date/time
DDTVAR3	computes the time-derivative of the requested variable at the specified date/time

Because it optimizes the interpolation problem for the user, INTERP3 is probably the most useful of these routines. An INTERP3 call to read/interpolate the variable HNO3 to 1230 GMT on February 4, 1995 is outlined below.

```
CHARACTER*16  FNAME, VNAME

REAL*4  ARRAY( NCOLS, NROWS, NLAYS )

...

IF ( .NOT. INTERP3('myfile','HNO3',1995035,123000,NCOLS*NROWS*NLAYS,ARRAY)) THEN

... (some kind of error happened--deal with it here)

END IF
```

With READ3 and XTRACT3, the user can use the "magic values" ALLVAR3' (= 'ALL', as defined in PARMS3.EXT) or ALLAYS3 (= -1, as also defined in PARMS3.EXT) as the variable name and/or layer number to read all variables or all layers from the file, respectively. For time independent files, the date and time arguments are ignored.

4.5. Writing Data Files in IO/API

CMAQ module developers should use the logical function WRITE3 to write data to files. For gridded, boundary, and custom files, the code may write either one time step of one variable at a time, or one entire time step of data at a time (in which case, use the "magic value" ALLVAR3 as the variable-name). For ID-referenced, profile, and grid-nest files, the code must write an entire time step at a time.

```
LOGICAL FUNCTION WRITE3( FNAME, VNAME, JDATE, JTIME, BUFFER)
```

where:	CHARACTER*(*) FNAME	file name for query
	CHARACTER*(*) VNAME	variable name (or ALLVAR3 (='ALL'))
	INTEGER JDATE	date, formatted YYYYDDD

INTEGER JTIME	time, formatted HHMMSS
BUFFER(*)	array holding output data

WRITE3 writes data for the variable with name VNAME, for the date and time (i.e., JDATE and JTIME) to an I/O API-formatted data file with logical name FNAME. For time independent files, JDATE and JTIME are ignored. If VNAME is the "magic name" ALLVAR3, WRITE3 writes all variables. If FNAME is a dictionary file, WRITE3 treats VNAME as a dictionary index (and ignores JDATE and JTIME). A typical WRITE3 call to write data for a given date and time might look like this:

```
REAL*4      ARRAY( NCOLS, NROWS, NLAYS, NVARS )

...

IF ( .NOT. WRITE3( 'myfile', 'HNO3', JDATE, JTIME, ARRAY ) ) THEN

...(some kind of error happened--deal with it here)

END IF

IF ( .NOT. WRITE3( 'afile', 'ALL', JDATE, JTIME, ARRAYB ) )  THEN

...(some kind of error happened--deal with it here)

END IF
```

4.6. CMAQ-Related I/O API Utilities

Data files in the CMAQ system can be easily manipulated by using the netCDF and I/O API utilities. Information regarding the most commonly employed utility routines is listed in Table 4.5, and they are further described in the following sections.

Table 4.5. CMAQ data manipulation utilities

Utility	Description
NCDUMP	print file as structured ASCII; from NCAR
M3XTRACT	extract a subset of variables from a file for a specified time interval
M3DIFF	compute statistics for pairs of variables
M3STAT	compute statistics for variables in a file
BCWNDW	build a boundary-condition file for a sub-grid window of a gridded file
M3EDHDR	edit header attributes/file descriptive parameters
M3TPROC	compute time period aggregates and write them to an output file
M3TSHIFT	copy/time shift data from a file
M3WNDW	window data from a gridded file to a sub-grid
M3FAKE	build a file according to user specifications, filled with dummy data
VERTOT	compute vertical-column totals of variables in a file
UTMTOOL	coordinate conversions and grid-related computations for Lat/Lon, Lambert, and UTM

4.6.1. NCDUMP

This program generates an ASCII representation of a netCDF file, either with or without an ASCII representation of the variable data in the file. NCDUMP was produced by the University Corporation for Atmospheric Research (the originators of netCDF) and is distributed by Unidata. This utility program is useful in determining the dimensions and contents of a MAQSIP file. The syntax for invoking ncdump in UNIX is the following

ncdump [-h] [-c] [-n name] [inputfile]

where:

-h produces only the "header" information in the output file (i.e., the declarations of dimensions, variables, and attributes, but no data value

-c produce the "header" information in the output file and the data values for coordinate variables

-n name

4.6.2. M3XTRACT

The M3XTRACT program extracts a variable or a selected subset of variables from a CMAQ file for a specified time period and a specified area and writes them to a separate CMAQ file. Optionally, the variables can be renamed during the process. Before running this program, or any other of the utilities, one is allowed to assign logical names to the physical file names of both files, according to IO/API conventions, using the operation

setenv <lname> <pname>

where lname equals the logical file name (e.g., INFILE), and

pname equals the physical file name (e.g., NC_195aa_cc3_g1)

The program will prompt the user for the logical names chosen for the input file and then extensively for the options to choose. The prompts will have default responses which can be accepted by hitting <RETURN>. The specification of starting date and time, target date and time, and duration for the run follow CMAQ date and time conventions.

This program is extremely useful in compressing large data files. For instance, an output concentration file containing 41 variables for a 16 × 23 × 26 domain occupies 226 Mbytes of disk space. The original concentration file can be saved to an archive while the analysis proceeds on a set of smaller files. Extracting just the ozone concentrations will result in a 5.5-Mbyte file. Further extracting only the surface layer ozone values will reduce the file size to 0.2 Mbytes. This utility is also useful for changing variable names within a CMAQ input/output file.

4.6.3. M3DIFF

The M3DIFF program performs comparison (and other) operations on pairs of variables from a user-specified pair of GRIDDED I/O API files and computes statistics for each specified variable and for the specified “difference” operation. An I/O API output file containing all the results of the specified operations can be optionally generated for viewing within scientific visualization tools.

setenv <infile A> <path name>

setenv <infile B> <path name>

setenv <reportfile> <path name>

m3diff [<infile A> <infile B> [<reportfile>]] [<DEFAULT>]

The default statistical analysis is a tabular listing (by time period) of the maximum value, minimum value, mean value, and the standard deviation of the 2 datasets. Several other calculations are also available:

Pointwise difference:	a-b
Pointwise difference:	b-a
Pointwise ratio:	a/b
Pointwise ratio:	b/a
Pointwise absolute value of difference:	|a-b|
Difference normalized by 1st grid:	(a-b)/a
Difference normalized by 2nd grid:	(a-b)/b
Difference normalized by 2nd grid:	(b-a)/b
Absolute value of difference normalized by A:	|a-b|/a
Absolute value of difference normalized by B:	|a-b|/b
Difference normalized by pointwise mean:	(a-b)/0.5(a+b)
Difference normalized by pointwise mean:	(b-a)/0.5(a+b)
Difference normalized by joint root mean square:	(a-b)/RMS(a&b)
Difference normalized by joint root mean square:	(b-a)/RMS(a&b)
Pointwise sum	a+b
Pointwise maximum	max(a,b)

Pointwise minimum	min(a,b)

4.6.4. M3STAT

The M3STAT program reports the result of a statistical analysis of user-selected variables in a CMAQ file. The default analysis identifies the maximum, minimum, mean, and standard deviations of the variable for individual time periods. The program can also be used to calculate the number and percentage of grid cells above a particular threshold. The following is a sample of m3stat output:

File: NC_195aa_cc3_g1.lyr1o3

Date and time:1995191:120000 (12:00:00 July 10, 1995)

Variable: O3

3-D grid statistics

Max 7.31641E-02 @(c,r,l)=(3,8,1)

Min 4.72329E-03 @(c,r,l)=(2,2,1)

Mean 4.47600E-02

Sigma 8.93855E-03

Number of times 6.00000E-02 exceeded: 18 fraction:0.48913043

Number of times 8.00000E-02 exceeded: 0 fraction:0.00000000

Number of times 1.00000E-01 exceeded: 0 fraction:0.00000000

Number of times 1.20000E-01 exceeded: 0 fraction:0.00000000

File: NC_195aa_cc3_g1.lyr1o3

Date and time:1995191:130000 (13:00:00 July 10, 1995)

Variable: O3

3-D grid statistics

Max 7.29948E-02 @(c,r,l)=(3,8,1)

Min 1.19016E-02 @(c,r,l)=(2,2,1)

Mean 5.12920E-02

Sigma 8.86664E-03

Number of times 6.00000E-02 exceeded: 54 fraction:1.46739140

Number of times 8.00000E-02 exceeded: 0 fraction:0.00000000

Number of times 1.00000E-01 exceeded: 0 fraction:0.00000000

Number of times 1.20000E-01 exceeded: 0 fraction:0.00000000

4.6.5. M3TPROC

The M3TPROC program computes time-aggregation for selected variables from time-stepped CMAQ files of types: BOUNDARY , CUSTOM , or GRIDDED. For each output time step and for each selected variable, the program writes the selected aggregate value of that variable to the output file, optionally renaming it in the process. For example, the program allows the user to construct a file of daily gridded 08:00-16:00 average ozone values derived from a standard MAQSIP output concentration file.

4.6.6. M3TSHIFT

The M3TSHIFT utility copies a CMAQ data file but allows user-specified changes to the time and date stamps on the files. This tool would prove useful in instances in which one data set was determined to be representative of multiple time periods.

4.6.7. M3WNDW

The M3WNDW program extracts a windowed set of data from a gridded CMAQ file and outputs this subset to a new gridded file for a specified time period. The definition of the sub-rectangle is user-specified by the entries of: LOCOL, LOROW, HICOL, and HIROW. One application of this utility would be to create an initial concentration file for a nested grid from the output of a parent grid.

4.6.8. BCWNDW

The BCWNDW program extracts a windowed set of data from a gridded CMAQ file and outputs this subset to a new BOUNDARY file for a specified time period. The definition of the sub-rectangle is user-specified by the entries of: LOCOL, LOROW, HICOL, and HIROW.

4.6.9. M3FAKE

The complex M3FAKE program prompts the user, one item at a time, for the contents of a GRIDDED, BOUNDARY, or CUSTOM file and then builds the file accordingly. It is a user-friendly means to create dummy data sets. In order to describe the horizontal and vertical coordinate systems, the user can specify the name of the grid as found in the GRIDDESC file or else may supply all needed parameters entry by entry.

4.6.10. UTMTOOL

The UTMTOOL program provides coordinate conversion back and forth among LAT-LON, UTM, and LAMBERT coordinate systems. Longitudes are specified in signed degrees such that for the United States, longitudes are negative. The coordinate system “UAM/EPS” refers to a grid with the kilometer units instead of the UTM standard units, which are meters.

UTMTOOL also provides grid-corner calculations, assuming either a UTM-based or a Lambert-based grid, as appropriate (to get UTM or Lambert corners of a LAT-LON based grid, work one corner at a time using the coordinate conversion facilities). The program will repeatedly prompt you for operation to be performed from the following menu:

1. Convert from LAT-LON to UTM

2. Convert from UTM to LAT-LON
3. Convert from UAM/EPS to LAT-LON
4. Initialize new LAMBERT projection by name
5. Define a new LAMBERT projection by angles
6. Convert from LAMBERT to UTM
7. Convert from UTM to LAMBERT
8. Convert from LAMBERT to LAT-LON
9. Convert from LAT-LON to LAMBERT
10. Get LAT-LON grid corners from UTM specs
11. Get UTM grid corners from LAT-LON specs
12. Get LAT-LON grid corners from LAMBERT specs
13. Get LAMBERT grid corners from LAT-LON specs
14. Quit the program.

4.7. References

Coats, C., 2005: The EDSS/Models-3 I/O API. [Available online at http://www.baronams.com/products/ioapi.]

Unidata, 2005: NetCDF. [Available onlie at http://www.unidata.ucar.edu/software/netcdf/.]

Chapter 5. CMAQ Programs and Libraries

Table of Contents

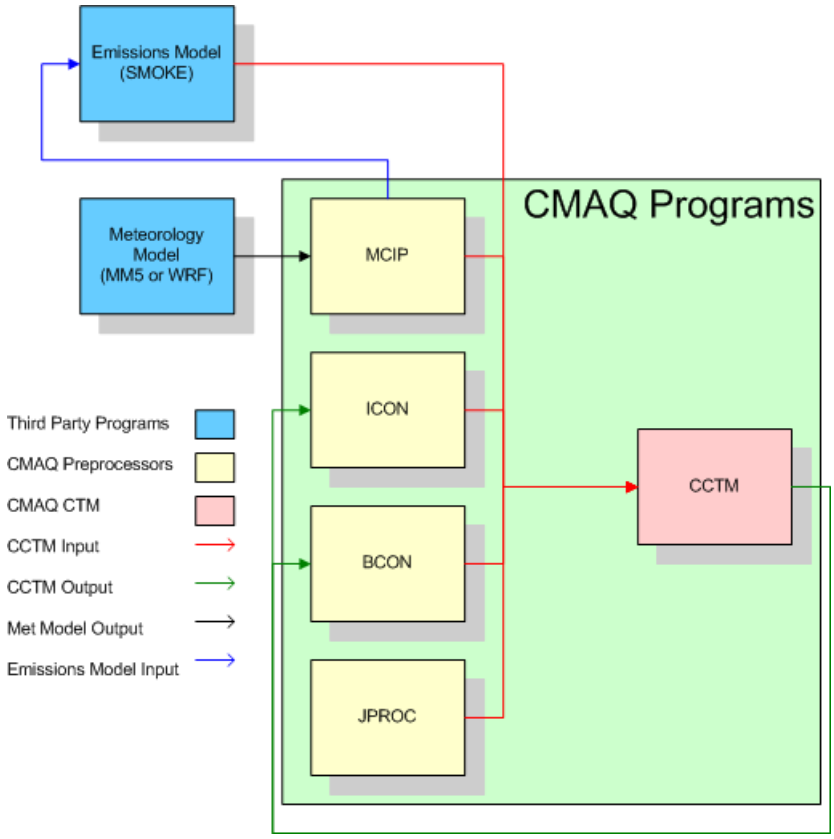
5.1. Overview 47
5.2. BCON 49
5.3. CCTM 55
5.4. CHEMMECH 66
5.5. ICON 69
5.6. JPROC 74
5.7. MCIP 78
5.8. PARIO 82
5.9. PDM 83
5.10. PROCAN 88
5.11. STENEX 95
5.12. References 96

5.1. Overview

CMAQ consists of aset of core programs that are needed to perform a basic air quality model simulation. Figure 5.1 shows these core programs and how they relate to each other. Utility programs such as PROCAN and PDM are excluded from this diagram; these programs are discussed later in this chapter. The lines between the boxes represent one or more files passing between the programs, and show the dependencies among the programs. The blue boxes represent programs that are not part of the CMAQ distribution package but supply data necessary for an air quality simulation. The yellow boxes represent the basic CMAQ preprocessors, MCIP, ICON, BCON, and JPROC. The red box represents the CMAQ chemistry-transport model, CCTM, the Eulerian air quality modeling component of CMAQ. Data flows between the CMAQ programs are represented in Figure 5.1 by arrows. The red arrows illustrate the flow of data from the CMAQ preprocessors to the CCTM. The green arrows show the data feedbacks from the CCTM to create initial and boundary conditions for nested simulations. The black arrow illustrates the connection between the meteorology model and MCIP. Finally, the blue arrow shows that the output from MCIP can be used to drive an emissions model, such as SMOKE.

Along with these programs, CMAQ also contains several utilities and two code libraries, STENEX and PARIO, to implement source specific and diagnostic features in CMAQ and for parallel applications of the model.

Figure 5.1. CMAQ core programs



The Emissions Model and Meteorology Model (Figure 5.1) are described in detail in Chapters7 and 8. An emissions model is required to convert annual, county-level emissions estimates to gridded, hourly emissions formatted for CMAQ. The SMOKE and CONCEPT models are two programs available for preparing emissions for CMAQ. Meteorology models, such as MM5 and WRF-ARW, generate gridded meteorology for input to both CMAQ and the emissions models.

MCIP is the first program in the CMAQ distribution package that a user should run when setting up a new simulation. It is used to preprocess the data from a meteorology model for CMAQ; the GRIDDESC file output from MCIP is needed to define the modeling grid.

ICON creates a binary netCDF initial conditions file for input to the CCTM. With the option of creating initial conditions from either a text file of vertical concentration profiles or from an existing CCTM output file, ICON outputs initial conditions data that are configured for a specific modeling grid and chemical parameterization.

BCON creates a binary netCDF lateral boundary conditions file for input to the CCTM. With the option of creating initial conditions from either a text file of vertical concentration profiles or from an existing CCTM output file, BCON outputs boundary conditions data that are configured for a specific modeling grid and chemical parameterization. If derived from an existing CCTM or larger-scale (e.g., global-scale) output file, BCON produces dynamic boundary conditions that vary in time and space. When derived from vertical concentration profiles, BCON produces static boundary conditions for input to the CCTM.

JPROC converts physical information about photoreactive molecules into clear-sky photolysis rate look up tables for input to the CCTM.

The CCTM is run last in the sequence of programs. All of the other CMAQ programs, emissions, and meteorology models are used to prepare the inputs to the CCTM. By using data that are synchronized for a particular modeling time

period, model grid, vertical layer configuration, and chemical parameterization, the CCTM produces hourly estimates of pollutant concentrations, wet and dry deposition rates, and visibility metrics.

In addition to the core programs shown in Figure 5.1, CMAQ also includes utilities and libraries for utilizing some of the special features in CMAQ and setting up the CCTM for multi-processor computing. CMAQ includes the PROCAN utility, for preparing process analysis simulations, CHEMMECH, the CMAQ chemical mechanism compiler, and the Plume Dynamics Model (PDM), for subgrid-scale representation of selected point sources (also known as plume-in-grid or "PinG"). The Stencil Exchange library (STENEX) is a module that the CCTM uses to control the communication between processors in a multiprocessor computing environment. Similarly, the CCTM uses the PARIO library to synchronize the reading and writing of information across and between multiple processors.

In the remaining sections of this chapter, we provide detailed descriptions of these programs, utilities and libraries, in alphabetical order. Information about the third party libraries used by CMAQ, such as the IO API, netCDF, and MPICH is available in Chapter 2.

5.2. BCON

5.2.1. Description

The program BCON prepares chemical boundary conditions for the CCTM from either ASCII vertical profiles, from an existing CCTM output concentration (CONC) file, or from a larger-scale (e.g., global) CTM. The boundary conditions created by BCON can be static in time and space (i.e. time-invariant with uniform concentrations in all boundary grid cells), dynamic in both time and space, or a combination of these two. The ASCII vertical profiles are primarily used to create static boundary conditions. Dynamic boundary conditions can be extracted from CONC files on either the same horizontal grid spacing (i.e., as a windowed modeling domain) or for a finer resolution model grid (i.e., for a nested simulation), or they can be interpolated from a larger-scale CTM simulation (which is analogous to defining lateral boundary conditions for MM5 or WRF-ARW).

There are two distinct modes of operation for BCON, and each depends on the nature of the input data. When creating BCON executables, the user must specify whether the input data will be ASCII vertical profiles or a CONC file by selecting either "profile" or "m3conc," respectively, for the setting of the *ModInpt* variable. This variable determines the input module to use when creating a BCON executable.

5.2.2. Files, configuration, and environment variables

Figure 5.2. BCON input and output files

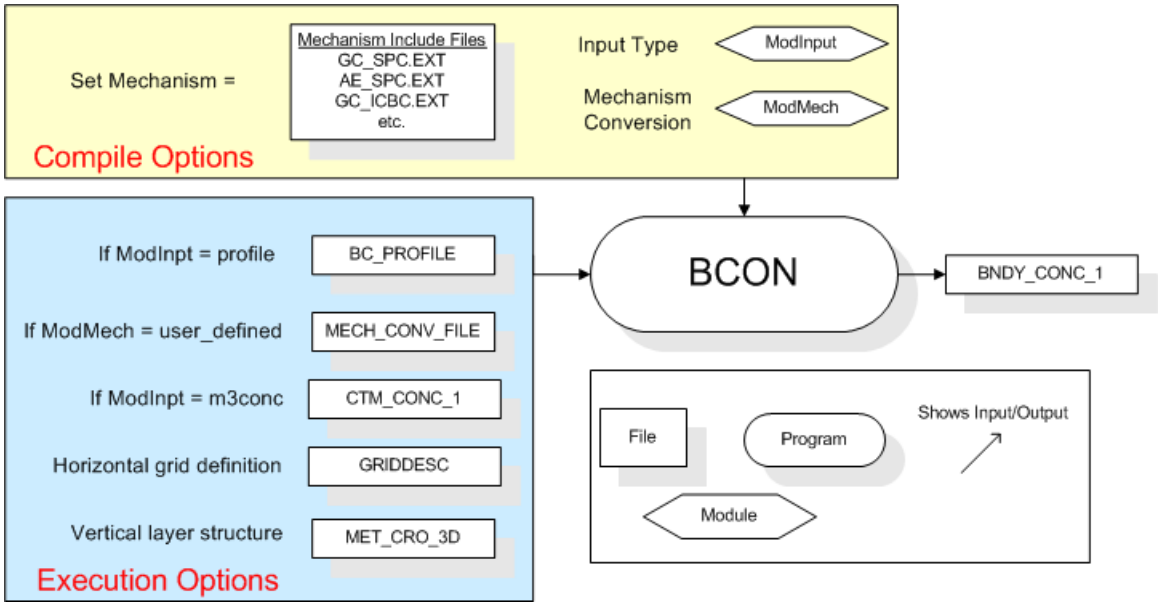


Figure 5.2 shows the input and output files for BCON. A distinction is made between the inputs that are invoked at compilation versus those invoked at execution of the program. When compiling BCON, the user specifies a chemical mechanism to determine the gas-phase chemistry and aerosol mechanism to calculate chemical boundary conditions for. Setting the Mechanism variable in the BCON compile script configures the program to use a specific set of mechanism include files to build an executable. Seperate BCON executables are hard-wired to a specific mechanism configuration.

At execution the user provides a data file of chemical conditions that BCON will convert to boundary conditions on a predefined model grid. Depending on how the user compiled the model, through the specification of the ModInpt variable, BCON will input either an ASCII vertical profile file (BC_PROFILE) or an existing CCTM concentration file (CTM_CONC_1). If the input file is not in the same chemical speciation as the simulation that the user is creating boundary conditions for, it is necessary to specify a chemical conversion option by setting the ModMech variable at compilation. The default conversion profiles in BCON are set up to convert from RADM2 speciation to either CB-IV, CB05, or SAPRC99 chemistry. It is possible to create a custom conversion file (MECH_CONV_FILE) and input this file to BCON by setting the ModMech variable to “user_defined” at compilation.

The horizontal grid and vertical layer structures for BCON are defined at execution through the input of a grid description (GRIDDESC) file and a meteorology cross-point 3-d (MET_CRO_3D) file, respectively. BCON interpolates between the input vertical layer structure and output layer structure if they are different.

5.2.2.1. BCON input files

Table 5.1. BCON input files

File Name	Format	Description
BC_PROFILE	ASCI	Vertical chemical profiles from which to derive boundary conditions; this file is created by the user; only used when BC is set to “profile”
MECH_CONV_FILE	ASCII	Mapping factors for converting between chemical mechanisms; this file is created by the user; only used when BCON is compiled with the ModMech configuration set to “user_defined”
CTM_CONC_1	GRDDED3	CMAQ concentration file from which to derive boundary conditions; this file is output from the CCTM; only used when BC is set to “m3conc”
CTM_PING_1	GRDDED3	Name and location of the plume-in-grid concentration file for the coarse domain; this file is output from the CCTM; only used when BCON_PING is set to “Y”
PING_PDM_1	GRDDED3	Name and location of the plume-in-grid plume dynamics file for the coarse domain; this file output from the PDM; only used when BCON_PING is set to “Y”
MET_CRO_3D_CRS	GRDDED3	Name and location of the course grid MET_CRO_3D file that is required for creating either plume-in-grid boundary conditions or if the vertical grid structure changes between nested simulations; this file is output by MCIP
MET_CRO_3D_FIN	GRDDED3	Name and location of the fine grid MET_CRO_3D file that is required if the vertical grid structure changes between nested simulations; this file is output by MCIP
GRIDDESC	ASCII	Horizontal grid description file for defining the model grid; this file is output by MCIP or can be created by the user
MET_CRO_3D	GRDDED3	3-D cross point meteorology file for defining the vertical layer structure of the model grid; this file is output by MCIP

5.2.2.2. BCON configuration options

The configuration options listed here are set during compilation of the BCON executable. When these options are invoked they create a binary executable that is fixed to the specified configuration. To change these options it is necessary to recompile BCON and create a new executable.

- Defines the action to be taken by the program M3BLD when extracting source code from CVS and compiling an executable.
  - *compile_all:* force compile, even if all the object files are current
  - *clean_up:* remove all source files upon successful compilation
  - *no_compile:* do everything except compile
  - *no_link:* do everything except link
  - *one_step:* compile and link in one step
  - *parse_only:* checks configuration file syntax
  - *show_only:* shows requested commands but does not execute them
  - *verbose:* shows requested commands as they are executed
- MakeOpt:

  Uncomment to build a Makfile to compile the executable.

- ModInpt:

  Defines the format of the boundary conditions input files to be used by BCON.
  - *m3conc*: input a CCTM CONC file; used for nested simulations or windows of a parent domain
  - *profile*: input an ASCII vertical profiles file
- ModMech:

  Defines if the input boundary conditions data need to be converted from one chemical mechanism to another.
  - *mc_noop*: do not perform any mechanism conversion; used when extracting boundary conditions from a CCTM CONC file for a nested simulation or when the input profiles are already formatted for the correct mechanism
  - *user_defined*: input the file defined by the MECH_CONV_FILE variable; used for custom mechanism conversions
  - *radm2_to_cb4*: convert the input boundary conditions profiles from RADM2 to CB-IV speciation
  - *radm2_to_cb05*: convert the input boundary conditions profiles from RADM2 to CB-05 speciation
  - *radm2_to_saprc99*: convert the input boundary conditions profiles from RADM2 to SAPRC99 speciation
- Mechanism:

Specifies the gas-phase, aerosol, and aqueous phase chemical mechanisms to create boundary conditions for. The choices for th
Mechanism variable are the mechanism directory names under the $M3MODEL/include/release directory. Examples include:

- *cb4_ae3_aq*: CB-IV gas-phase mechanism, 3rd generation CMAQ aerosol mechanism, aqueous/cloud chemistry
- *cb4_ae4_aq*: CB-IV gas-phase mechanism, 4th generation CMAQ aerosol mechanism, aqueous/cloud chemistry
- *cb05_ae4_aq*: CB-05 gas-phase mechanism, 4th generation CMAQ aerosol mechanism, aqueous/cloud chemistry
- *saprc99_ae4_aq*: SAPRC99 gas phase mechanism, 4th generation CMAQ aerosol mechanism, aqueous/cloud chemistry
- *saprc99_ae3_aq*: SAPRC99 gas phase mechanism and 3rd generation CMAQ aerosol mechanism, aqueous/cloud chemistry

5.2.2.3. BCON compilation

Section 3.3 provides an overview of how to install and compile the CMAQ programs for the benchmark simulation. Follow the steps outlined in Section 3.3 to compile new versions of BCON:

1. If not already done, build the CMAQ source code and compilation management program (M3BLD); this only needs to be done the first time CMAQ is installed
2. Install and compile the I/O API and netCDF libraries; if these are already available from a previous CMAQ compilation, configure the BCON build script to use the available libraries
3. Configure the BCON build script for your application
4. Invoke the build script to create an executable:

```
./bldit.bcon.pgf
```

5.2.2.4. BCON execution options

The environment variables listed here are invoked during execution of the program and are set in the BCON run script.

- EXEC: [default: BCON_${CFG}]

  Executable to use for the simulation

- NPCOL_NPROW: [default: 1 1]

  Domain decomposition for parallel mode; BCON is normally run in a single processor environment, so this setting should alwa

- GRIDDESC: [default: ../GRIDDESC1]

  Grid description file for setting the horizontal grid definition

- GRID_NAME:

  Name of the grid definition contained in the GRIDDESC file that specifies the horizontal grid for the current application of the

- LAYER_FILE:

  Name and location of a MET_CRO_3D file for specifying the vertical layer structure for the current application of the model

- OUTDIR: [default: $M3DATA/bcon]

  Output data directory

- BC:

  Sets the input file type. The setting of this variable determines how the input and output environment variables are set by the run script.

  - *profile:* sets the output file name to include the tag “profile” in the name; uses the variable BC_PROFILE to point to an ASCII vertical profile file for input to BCON. Also optionally looks for the variable MECH_CONV_FILE to point to a user defined mechanism conversion file.
  - *m3conc:* used for nested simulations, sets the output file name to include a start date in the name; uses the variable CTM_CONC_1 to point to a CCTM CONC file for input to BCON. Also looks for optional Plume-in-Grid inputs if the BCON_PING variable is set to “YES”.

- DATE:

  Sets the Julian date to use for tagging the BCON output file for nested runs.

- BCON_PING:

  Flag controlling the creation of boundary conditions for a plume-in-grid simulation.

  - *YES*: Use plume-in-grid input files
  - *NO*: Do not use plume-in-grid input files

- SDATE:

  Julian start date for extracting boundary conditions from a CCTM CONC file for a nested simulation. If SDATE is not set, it will be set automatically from the CTM_CONC_1 file.

- STIME

  Start time for extracting boundary conditions from a CCTM CONC file for a nested simulation. If STIME is not set, it will be set automatically from the CTM_CONC_1 file.

- RUNLEN

  Number of hours of boundary conditions to extract from a CCTM CONC file for a nested simulation. If RUNLEN is not set, it will be set automatically from the CTM_CONC_1 file.

5.2.2.5. BCON output files

Table 5.2. BCON output files

File Name	Format	Description
BNDY_CONC_1	BNDARY3	Name and location of the gridded boundary conditions data output on the model grid defined by GRID_NAME

The default location of the **BCON** output files is the $M3DATA/bcon directory, controlled by the OUTDIR variable in the run script. The default naming convention for all **BCON** output files uses the APPL and GRID_NAME environment variables in the file name. For boundary conditions created from existing CCTM CONC files, the Julian date is also used in the file name through the DATE environment variable. All of the file naming variables for **BCON** outputs are set in the run script.

5.3. CCTM

5.3.1. Description

The program **CCTM** is the CMAQ chemistry-transport model, the Eulerian air quality model component of CMAQ. Typically run last in the sequence of the CMAQ programs, **CCTM** uses input data produced by all of the other CMAQ programs and from an emissions model, such as SMOKE. **CCTM** produces multiple output files for each simulation. The basic **CCTM** outputs include instantaneous and average hourly concentration files, wet and dry deposition files, and visibility estimates. Other **CCTM** outputs can include diagnostic aerosol and cloud files, hourly plume-in-grid files, and processes analysis files.

CCTM contains several science configurations for simulating transport, chemistry, and deposition. All of the science configuration options, including the chemical mechanism, in CCTM are set when compiling the executable. The model grid and vertical layer structure for CCTM are set at execution. The important distinction between selecting the science configuration and the model grid/layer configuration is that CCTM does not need to be recompiled when changing model grids/layers but does need to be recompiled when new science options are invoked.

Optional output files are created when their associated optional processes are invoked in CCTM.

5.3.2. Files, configuration, and environment variables

[Insert Schematic of CCTM]

5.3.2.1. CCTM input files

Table 5.3. CCTM input files

File Name	Format	Description
OCEAN_1	GRDDED3	Name and location of the time-independent 2-D file for defining the fraction of each model grid cell covered by open ocean
EMIS_1	GRDDED3	Name and location of the time-dependent 2-D or 3-D emission file speciated for a particular gas-phase chemical mechanism and PM model; output from an emission model, such as SMOKE or CONCEPT
MEPSE_1	GRDDED3	Name and location of the major emitting point sources emissions file, required only when the CCTM is compiled for PinG simulations; output from an emission model, such as SMOKE or CONCEPT
PDM_PING_1	GRDDED3	Name and location of the active plume file, required only when the CCTM is compiled for PinG simulations; output from the PDM
INIT_[GASC|AERO| NONR|TRAC]_1	GRDDED3	Name and location of the time-dependent, single time step, 3-D initial conditions file speciated for a particular gas-phase chemical mechanism and PM model; output from ICON
BNDY_[GASC|AERO| NONR|TRAC]_1	GRDDED3	Name and location of the time-dependent, either single time step or multi-time step, 3-D boundary conditions file speciated for a particular gas-phase chemical mechanism and PM model; output from BCON
GRID_CRO_2D	GRDDED3	Name and location of the time-independent 2-D cross point meteorology file; output by MCIP
GRID_DOT_2D	GRDDED3	Name and location of the time-independent 2-D dot point meteorology file; output by MCIP
MET_CRO_2D	GRDDED3	Name and location of the time-dependent 2-D cross point meteorology file
MET_DOT_3D	GRDDED3	Name and location of the time-dependent 3-D dot point meteorology file; output by MCIP
MET_CRO_3D	GRDDED3	Name and location of the time-dependent 3-D cross point meteorology file; output by MCIP

File Name	Format	Description
MET_BDY_3D	BNDARY3	Name and location of the time-dependent 3-D boundary meteorology file; output by MCIP
XJ_DATA	ASCII	Name and location of the daily clear sky photolysis rates file speciated for a particular gas-phase chemical mechanism; output from JPROC

5.3.2.2. CCTM compilation options

The configuration options listed here are set during compilation of the **CCTM** executable. When these options are invoked they create a binary executable that is fixed to the specified configuration. The change these options it is necessary to recompile **CCTM** and create a new executable.

- Opt: [default: verbose]

  Defines the action to be taken by the program **M3BLD** when extracting source code from CVS and compiling an executable.

  - *compile_all*: force compile, even if all the object files are current
  - *clean_up*: remove all source files upon successful compilation
  - *no_compile*: do everything except compile
  - *no_link*: do everything except link
  - *one_step*: compile and link in one step
  - *parse_only*: checks configuration file syntax
  - *show_only*: shows requested commands but does not execute them
  - *verbose*: shows requested commands as they are executed

- MakeOpt:

  Uncomment to build a Makefile to compile the executable.

- ParOpt:

  Uncomment to build an executable for running on multiple processors. Invoking this command requires the availability of a parallel Stenex library file, a Pario library file, and an MPI library/include files.

- ModDriver: [default: ctm_yamo]

  The CCTM general coordinate driver module.

  - *ctm*: air density-based scheme for mass-conserving advection
  - *ctm_yamo*: Yamartino scheme for mass-conserving advection

- ModInit: [default: init_yamo]

  The CCTM time-step initialization module.

  - *init*: air density-based scheme for mass-conserving advection
  - *init_yamo*: Yamartino scheme for mass-conserving advection

- ModAdjc: [default: // Yamartino option]

  Mass conservation error adjustment scheme. Corrects for mass inconsistencies arising from how the input meteorology treats density and wind fields. This adjustment is only required if the air density-based scheme for mass-conserving advection is selected.

  - *adjcon_noop*: deactivate the air density-based conservation adjustment scheme
  - *denrate*: adjust the vertical advection error term from the meteorology model by air density from the CCTM
  - //: comment out for Yamartino scheme for mass-conserving advection

- ModCpl: [default: gencoor]

  Unit conversion and concentration coupling module. The only option is for the CMAQ generalized coordinate (gencoor).

- ModHadv: [default: hyamo]

  Horizontal advection module

  - *hadv_noop*: deactivate horizontal advection
  - *hppm*: use the unmodified Piecewise Parabolic Method to calculate horizontal advection
  - *hyamo*: use the global mass-conserving scheme to calculate horizontal advection

- ModVadv: [default: vyamo]

  Vertical advection module

  - *vadv_noop*: deactivate vertical advection
  - *vppm*: use the unmodified Piecewise Parabolic Method to calculate vertical advection
  - *vyamo*: use the global mass-conserving scheme to calculate vertical advection

- ModHdiff: [default: multiscale]

  Horizontal diffusion module

  - *hdiff_noop*: deactivate horizontal diffusion
  - *multiscale*: diffusion coefficient based on local wind deformation

- ModVdiff: [default: acm2]

  Vertical diffusion module

  - *vdiff_noop*: deactivate vertical diffusion
  - *eddy*: calculate vertical diffusion using eddy diffusivity theory
  - *eddy_carb*: eddy vertical diffusion instrumented for carbonaceous aerosol apportionment

- *edd_hg*: eddy vertical diffusion instrumented for the mercury CMAQ model
- *eddy_sulf*: eddy vertical diffusion instrumented for the sulfate tracking model
- *eddy_tx*: eddy vertical diffusion instrumented for the air toxics CMAQ model
- *acm2*: calculate vertical diffusion using the Asymmetric Convective Model version 2
- *acm2_carb*: ACM2 vertical diffusion instrumented for carbonaceous aerosol apportionment
- *acm2_hg*: ACM2 vertical diffusion instrumented for the mercury CMAQ model
- *acm2_sulf*: ACM2 vertical diffusion instrumented for the sulfate tracking model
- *acm2_tx*: ACM2 vertical diffusion instrumented for the air toxics CMAQ model

- ModPhot: [default: phot]

  Photolysis calculation module

  - *phot_noop*: deactivate photolysis rate calculations
  - *phot*: calculate photolysis rates

- ModPing: [default: ping_noop]

  Plume in Grid module

  - *ping_noop*: deactivate plume in grid model
  - *ping_smvgear_aero3*: use the CMAQ plume dynamics model with an internal SMVGEAR chemistry solver for gas-phase chemistry with aero3
  - *ping_smvgear_aero4*: use the CMAQ plume dynamics model with an internal SMVGEAR chemistry solver for gas-phase chemistry with aero4

- ModChem: [default: ebi_cb05]

  Gas-phase chemistry solver module

  - *chem_noop*: deactivate gas-phase chemistry
  - *smvgear*: use the SMVGEAR chemistry solver
  - *ros3*: use the Rosenbrock chemistry solver
  - *ebi_cb4*: use the Euler Backward Iterative solver optimized for the Carbon Bond-IV mechanism
  - *ebi_cb4cl*: use the Euler Backward Iterative solver optimized for the Carbon Bond-IV mechanism with chlorine chemistry
  - *ebi_cb4hg*: use the Euler Backward Iterative solver optimized for the Carbon Bond-IV mechanism with mercury
  - *ebi_cb4tx1p*: use the Euler Backward Iterative solver optimized for the Carbon Bond-IV mechanism with air toxics
  - *ebi_cb05*: use the Euler Backward Iterative solver optimized for the Carbon Bond-05 mechanism

  - *ebi_cb05hg*: use the Euler Backward Iterative solver optimized for the Carbon Bond-05 mechanism with mercury
  - *ebi_cb05cltx*: use the Euler Backward Iterative solver optimized for the Carbon Bond-05 mechanism with air toxics
  - *ebi_saprc99*: use the Euler Backward Iterative solver optimized for the SAPRC-99 mechanism
  - *ebi_saprc99tx2p*: use the Euler Backward Iterative solver optimized for the SAPRC-99 mechanism with air toxics
  - *ebi_saprc99tx3*: use the Euler Backward Iterative solver optimized for the SAPRC-99 mechanism with air toxics
- ModAero: [default: aero4]

  CMAQ aerosol module

  - *aero_noop*: deactivate aerosol chemistry
  - *aero3*: the 3rd generation modal CMAQ aerosol model
  - *aero3_carb*: aero3 instrumented with primary carbonaceous aerosol apportionment
  - *aero3_sulf*: aero3 instrumented with a sulfate tracking model
  - *aero3_hg*: aero3 with mercury chemistry
  - *aero4*: the 3rd generation modal CMAQ aerosol model with extensions for sea salt emissions and thermodynamics
  - *aero4_carb*: aero4 instrumented with primary carbonaceous aerosol apportionment
  - *aero4_sulf*: aero4 instrumented with a sulfate tracking model
  - *aero4_hg*: aero4 with mercury chemistry
  - *aero4_tx*: aero4 with hazard air pollutant aerosol treatment
- ModAdepv: [default: aero_depv2]

  CMAQ aerosol deposition velocity module

  - *aero_depv_noop*: deactivate aerosol deposition
  - *aero_depv2*: 2nd generation CMAQ aerosol deposition velocity routine
  - *aero_depv2_carb*: aero_depv2 instrumented for carbonaceous aerosol apportionment
  - *aero_depv2_hg*: aero_depv2 instrumented for mercury modeling
  - *aero_depv2_sulf*: aero_depv2 instrumented for the sulfate tracking model
- ModCloud: [default: acm]

  CMAQ cloud module for modeling the impacts of clouds on deposition, mixing, photolysis, and aqueous chemistry

  - *cloud_noop*: deactivate clouds in modeling

- *cloud_radm*: RADM cloud processor
- *cloud_radm_sulf*: RADM cloud processor instrumented for the sulfate tracking model
- *cloud_acm*: RADM-based cloud processor that uses the asymmetric convective model to compute convective mixing
- *cloud_acm_hg*: ACM cloud model instrumented for modeling mercury
- *cloud_acm_tx*: ACM cloud model instrumented for modeling air toxics

- ModPa: [default: pa]

  Process analysis module

  - *pa*: only configuration option at the module level; to turn process analysis on/off in CMAQ use the PAOpt variable (see below).

- ModUtil: [default: util]

  CMAQ utility modules

  - *util*: only configuration option at the module level

- Mechanism: [default: cb05_ae4_aq]

  Specifies the gas-phase, aerosol, and aqueous phase chemical mechanisms to use for modeling air quality. The choices for the Mechanism variable are the mechanism directory names under the $M3MODEL/include/release directory. Examples include:

  - *cb4_ae3_aq*: CB-IV gas-phase mechanism, 3rd generation CMAQ aerosol mechanism, aqueous/cloud chemistry
  - *cb4_ae4_aq*: CB-IV gas-phase mechanism, 3rd generation CMAQ aerosol mechanism with sea salt, aqueous/cloud chemistry
  - *cb05_ae4_aq*: CB-05 gas-phase mechanism, 4th generation CMAQ aerosol mechanism with sea salt, aqueous/cloud chemistry
  - *saprc99_ae4_aq*: SAPRC99 gas phase mechanism, 3rd generation CMAQ aerosol mechanism with sea salt, aqueous/cloud chemistry
  - *saprc99_aq*: SAPRC99 gas phase mechanism and aqueous/cloud chemistry
  - *cb4ae4_ca_aq*: CB-IV gas-phase mechanism, 4th generation CMAQ aerosol mechanism with sea salt, aqueous/cloud chemistry, carbonaceous aerosol apportionment
  - *saprc99_ae4st_aq*: SAPRC99 gas phase mechanism, 4th generation CMAQ aerosol mechanism, aqueous/cloud chemistry, sulfate tracking model

- PAOpt: [default: pa_noop]

  Specifies the process analysis configuration to use for CMAQ. The choices for the PAOpt variable are the available directories for process analysis include files under the $M3MODEL/include/release directory.

5.3.2.3. CCTM compilation

Section 3.3 provides an overview of how to install and compile the CMAQ programs for the tutorial simulation. Follow the steps outlined in Section 3.3 to compile new versions of CCTM:

1. If you have not already done so, build the CMAQ source code and compilation management program (**M3BLD**); this only needs to be done the first time you install CMAQ

2. Install and compile the I/O API, netCDF, and MPICH libraries; if these are already available from a previous CMAQ compilation, configure the **CCTM** build script to use the available libraries

3. If you have not already done so, compile the STENEX and PARIO libraries

4. Configure the **CCTM** build script for your application

5. Invoke the build script to create an executable:

   ./bldit.cctm.pgf

5.3.2.4. CCTM execution options

The environment variables listed here are invoked during execution of the program and are set in the CCTM run script.

- EXEC: [default: CCTM_${CFG}]
  Executable to use for the simulation

- NPCOL_NPROW: [default: 1 1]
  Domain decomposition for parallel mode; recommended configuration is for the number of columns to be larger than the number of rows. For example, if running with 8 processors, the recommended setting is “4 2”.

- NPROCS: [default: 1]
  Number of processors for parallel execution; equal to the product of NPCOL x NPROW

- STDATE:
  Simulation start date (YYYYDDD)

- STIME
  Simulation start time (HHMMSS)

- NSTEPS: [default: 240000]
  Number of simulation time steps (HHMMSS)

- TSTEP: [default: 010000]
  Simulation output time step interval (HHMMSS)

- LOGFILE: [default: $BASE/$APPL.log]
  Uncomment to capture CCTM standard output to a log file; the LOGFILE variable sets the name and location of the log

- IOAPI_LOG_WRITE: [default: F]
  [T|F]; set to “F” to turn off excess WRITE3 logging by the I/O API

- FL_ERR_STOP: [default: F]
  [T|F|Y|N]; set to “F” or “N”

- DISP: [default: keep]
  Controls the maintenance of existing log files.
  - *delete*: delete output log if it already exists
  - *keep*: abort simulation if output log exists
- OUTDIR: [default: $M3DATA/cctm]
  CCTM output file directory location
- CTM_APPL: [default: $APPL]
  CCTM log file naming extension
- GRIDDESC: [default: ../GRIDDESC1]
  Grid description file for setting the horizontal grid definition
- GRID_NAME:
  Name of the grid definition contained in the GRIDDESC file that specifies the horizontal grid for the current application of the model
- AVG_CONC_SPCS: [default: “O3 NO NO2 ASO4I ASO4J NH3”]

  Model species for calculating integral average concentrations for each output time step. Options can be any of the standard output species that are written to the CCTM CONC file. The species in this list will be written to the ACONC output file.
- ACONC_BLEV_ELEV: [default: “1 1”]

  Vertical model layer range for integral average concentrations; this variable sets the lower and upper layers over which to calculate integral average concentrations.
- OCEANpath: [default: $M3DATA/emis/tut02]
  Directory path for the input sea salt emissions file.
- EMISpath: [default: $M3DATA/emis/tut02]
  Directory path for the input emissions file
- TR_EMpath: [default: commented out]
  Directory path for the input tracer emissions file
- [GC|AE|NR|TR]_ICpath
  Directory path for the input initial conditions file
- [GC|AE|NR|TR]_BCpath
  Directory path for the input boundary conditions file
- METpath
  Directory path for the input meteorology files
- JVALpath
  Directory path for input clear-sky photolysis rate files

5.3.2.5. CCTM output files

Table 5.4. CCTM output files

File Name	Format	Description
CTM_CONC_1	GRDDED3	Name and location of hourly 3-D instantaneous gas- and aerosol-phase pollutant estimates
A_CONC_1	GRDDED3	Name and location of hourly 2-D or 3-D integral average gas- and aerosol-phase pollutant estimates
CTM_DRY_DEP_1	GRDDED3	Name and location of hourly 3-D gas- and aerosol-phase dry deposition estimates
CTM_WET_DEP_1	GRDDED3	Name and location of hourly 3-D gas- and aerosol-phase wet deposition estimates
CTM_VIS_1	GRDDED3	Name and location of hourly 3-D visibility metrics
CTM_DIAM_1	GRDDED3	Name and location of hourly 3-D aerosol diagnostics; dp and sigmas for Aitken and Accumulation mode aerosol species; set the variable CTM_AERODIAG to “T” in the CCTM run script to write this additional file
CTM_PING_1	GRDDED3	Name and location of hourly 3-D residual plume mass for Plume in Grid sources
CTM_PING_DIAM_1	GRDDED3	Name and location of hourly 3-D aerosol diagnostics for Plume in Grid sources
CTM_PING_DDEP_1	GRDDED3	Name and location of hourly 3-D gas- and aerosol-phase dry deposition estimates for Plume in Grid sources
CTM_IPR_[1-3]	GRDDED3	Name and location of hourly 2-D or 3-D integrated process rate files; multiple files written when the CCTM is configured to track a large number of process/pollutant combinations
CTM_IRR_[1-3]	GRDDED3	Name and location of hourly 2-D or 3-D integrated reaction rate files; multiple files written when the CCTM is configured to track a large number of reaction/pollutant combinations

The default location of the **CCTM** output files is the $M3DATA/cctm directory, controlled by the OUTDIR variable in the run script. The default naming convention for all **CCTM** output files uses the EXEC and APPL environment variables in the file name. All of the file naming variables for **CCTM** outputs are set in the run script.

5.4. CHEMMECH

5.4.1. Description

The program **CHEMMECH** generates mechanism include files for all chemical mechanism-dependent CMAQ programs. Using an ASCII mechanism definition file as input, a combination of the Fortran 77 program **CHEMMECH** and the Python script, include_generator.py, created by Chao-Jung Chien at UC-Riverside, creates all of the Fortran include files that define the gas-phase chemical mechanisms for the CMAQ programs.

To implement new mechanisms created by **CHEMMECH** in the CMAQ programs, manually move the output include files from **CHEMMECH** to the $M3MODEL/include/release/{NewMechanism} directory within the CMAQ installation directories, where "NewMechanism" corresponds to the name of the new mechanism created with **CHEMMECH**. To invoke this new mechanism, set the Mechanism variable in the build script for the applicable CMAQ program to the name of the mechanism directory that you created. For example, if you used **CHEMMECH** to create include files for a mechanism that you call cb4_2006, create a directory under $M3MODEL/include/release called cb4_2006, move the include files created by **CHEMMECH** to this directory, and compile the CMAQ programs for this new mechanism by setting the Mechanism variable within the build scripts to cb4_2006.

5.4.2. Files, configuration, and environment variables

Figure 5.3. CHEMMECH input and output files

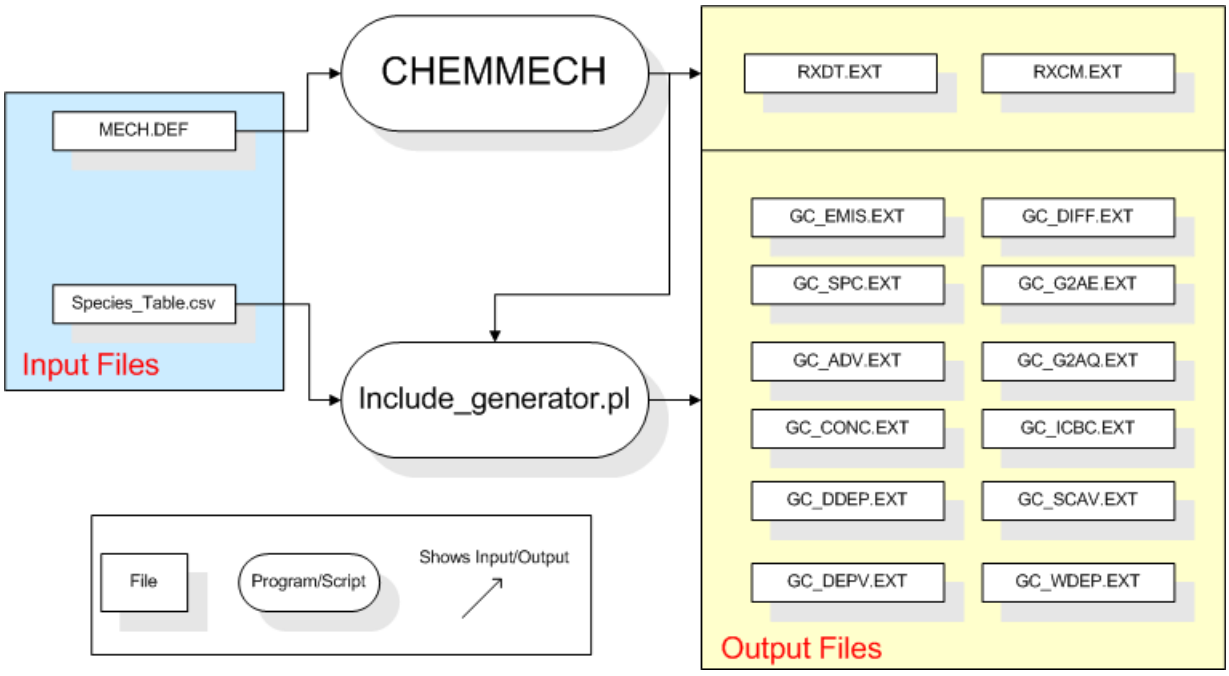


Figure 3 shows the input and output files and configuration options for **CHEMMECH**. The full set of mechanism include files required by the CMAQ programs are generated in two steps. In the first step, the program **CHEMMECH** is run with the mechanism definition file, MECH.DEF provided as input. The resulting RXDM.EXT and RXCM.EXT include files are then input to the Python script, **include_generator.py**, along with the Species_Table.csv file to create the full set of include files needed to compile CMAQ with a new chemical mechanism configuration. The Species_Table.csv file defines the atmospheric processes that each species undergoes in the model and is generated by the Excel spreadsheet Species_Table.xls that is distributed with **CHEMMECH**.

To implement a new mechanism in CMAQ edit a MECH.DEF file that is associated with a base mechanism and provide this new MECH.DEF files as input to **CHEMMECH**. Edit the default run script MP.saprc99.csh to point to the new MECH.DEF file. This script will call both the **CHEMMECH** executable and the **include_generator.py** script to create the full set of mechanism include files required by the CMAQ programs. There are no horizontal grid, vertical layer, or temporal settings for **CHEMMECH**.

5.4.2.1. CHEMMECH input files

Table 5.5. CHEMMECH input files

File Name	Format	Description
MCFL (MECH.DEF)	ASCII	CMAQ mechanism definition file; photochemical mechanism listing with both mechanistic and kinetic information about all reactions that compose a chemical mechanism
SPST	ASCII/CSV	Species table; comma delimited file generated from an Excel spreadsheet that lists the atmospheric processes that affect each model species

5.4.2.2. CHEMMECH compilation options

The configuration options listed here are set during compilation of the **CHEMMECH** executable. The compiler-specific options, such as the compiler names and flags, are set in the **make.reader** script distributed with the program. The directory paths of the **CHEMMECH** source code and working directory are set in the **makeit** script.

- FC:[default: pgf90]
  Fortran compiler path and name
- FFLAGS:
  Fortran compilation flags
- CC:[default: cc]
  C compiler path and name
- C_FLAGS:
  C compilation flags
- BASE: [default: $cwd]
  Working directory for compiling **CHEMMECH**
- SRC: [default: ../]
  Location of **CHEMMECH** source code

5.4.2.3. CHEMMECH compilation

To compile the **CHEMMECH** program run the following command in the directory that contains the scripts:

```
./makeit MP
```

For the compilation to be successful you must have the Portland Group Fortran and Gnu C compilers added to your path. To port **CHEMMECH** to other compilers, change the compiler names, locations, and flags in the **make.reader** script.

5.4.2.4. CHEMMECH execution options

The environment variables listed here are invoked during execution of the program and are set in the **CHEMMECH** run script. The default run script is called MP.saprc99.csh.

- BASE: [default: $cwd]
  Working directory path
- Xpath: [default: $BASE]
  Executable directory path
- Mpath: [default: ../mech]
  Directory path to the MECH.DEF file
- Opath: [default: ../exts]
  Output file directory path
- APPL:
  Simulation identifier
- EXEC: [default: CHEMMECH]
  Executable name
- MCFL: [default: mech.def.saprc99]
  Mechanism definition file name
- EXSPCS: [default: $Opath/SPECIES.ext]
  Output file name when CHEMMECH is run in SP mode
- EXRXDT: [default: $Opath/RXDT.EXT]
  Name of output mechanism data include file
- EXRXCM: [default: $Opath/RXCM.EXT]
  Name of output mechanism common include file

5.4.2.5. CHEMMECH output files

Table 5.6. CHEMMECH output files

File Name	Format	Description
RXCM.EXT	ASCII	Mechanism common include file; lists all of the chemical mechanism variables and parameters
RXDT.EXT	ASCII	Mechanism data include file; chemical mechanism definition formatted as DATA blocks to be read in as CMAQ source code
GC_ADV.EXT	ASCII	File listing the gas-phase model species that are transported by advection
GC_CONC.EXT	ASCII	File listing the gas-phase model species to write to the CCTM output CONC file
GC_DDEP.EXT	ASCII	File listing the gas-phase model species that dry deposit
GC_DEPV.EXT	ASCII	File listing the gas-phase model species for which to calculate deposition velocities
GC_DIFF.EXT	ASCII	File listing the gas-phase model species that are transported by diffusion
GC_EMIS.EXT	ASCII	File listing the gas-phase model species that are emitted into the model
GC_G2AE.EXT	ASCII	File listing the gas-phase model species that react to form aerosols
GC_ICBC.EXT	ASCII	File listing the gas-phase model species that require initial and boundary conditions
GC_SCAV.EXT	ASCII	
GC_SPC.EXT	ASCII	File listing the gas-phase model species
GC_WDEP.EXT	ASCII	File listing the gas-phase model species that wet deposit

The default location of the **CHEMMECH** output files is the exts directory, controlled by the Opath variable in the run script. To compile a version of the CMAQ programs that use the include files created by **CHEMMECH**, these output include files need to be moved to a new directory under the $M3MODEL/include/release directory. Point the CMAQ build scripts to this new directory through the “Mechanism” variable.

5.5. ICON

5.5.1. Description

The program **ICON** prepares chemical initial conditions (ICs) for the CCTM from either ASCII vertical profiles or from an existing **CCTM** output concentration (CONC) file. **ICON** creates an output file with a single time step that

represents the chemical conditions in each grid cell at the beginning of a **CCTM** simulation. The ICs can be either spatially uniform or variable across the model grid, depending on the source of the initial chemical concentration data. If deriving ICs from the ASCII vertical profiles, **ICON** can only create spatially uniform ICs within each model layer; it can create different initial conditions across model layers. **ICON** can extract spatially varying ICs from CONC files on either the same grid cell resolution, as a windowed modeling domain, or for a finer resolution model grid, as for a nested simulation.

Two distinct modes of operation exist for **ICON**, depending on the nature of the input data. When creating **ICON** executables, the user must specify whether the input data will be ASCII vertical profiles or a CONC file by selecting either “profile” or “m3conc”, respectively, for the setting of the ModInpt variable. This variable determines the input module to use when creating a **ICON** executable.

5.5.2. Files, configuration, and environment variables

Figure 5.4. ICON input and output files

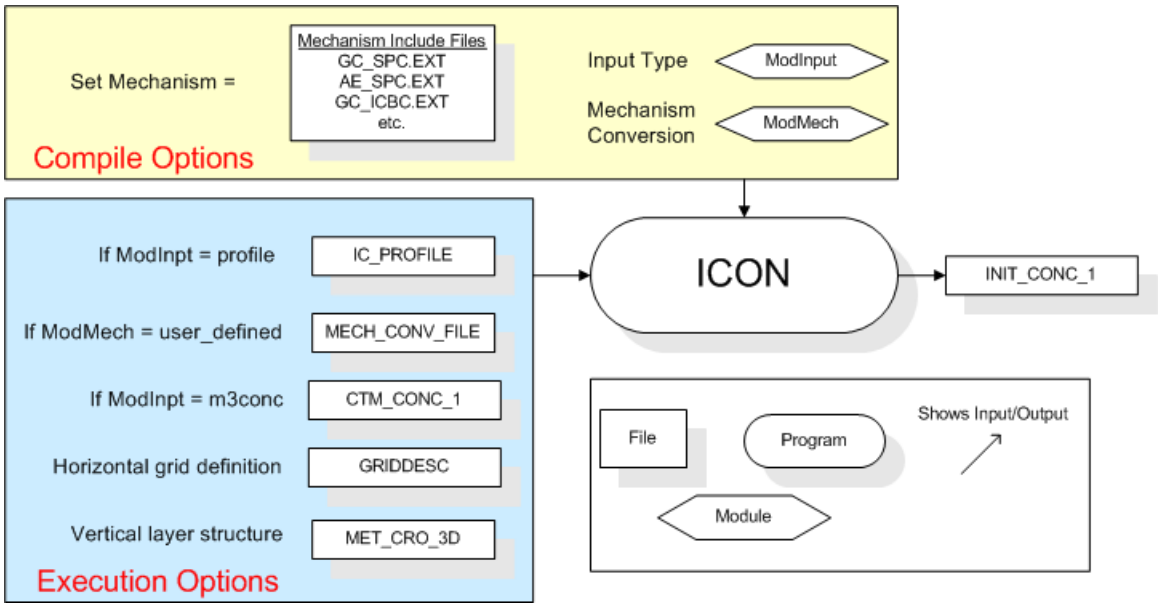


Figure 3 shows the input and output files and configuration options for **ICON**. A distinction is made between the options that are invoked at compilation versus those invoked at execution of the program. When compiling **ICON**, the user specifies a chemical mechanism to determine the gas-phase chemistry and aerosol mechanism for which to calculate chemical ICs. Setting the Mechanism variable in the **ICON** compile script configures the program to use a specific set of mechanism include files to build an executable. **ICON** executables are hard-wired to a specific mechanism configuration.

At execution the user provides a data file of chemical conditions that **ICON** will convert to ICs on a predefined model grid. Depending on how the user compiled the model, through the specification of the ModInpt variable, **ICON** will input either an ASCII vertical profile file (IC_PROFILE) or an existing CCTM concentration file (CTM_CONC_1). If the input file is not in the same chemical speciation as the simulation that the user is creating ICs for, it is necessary to specify a chemical conversion option by setting the ModMech variable at compilation. The default conversion profiles in **ICON** are set up to convert from RADM2 speciation to either CB-IV or SAPRC99 chemistry. It is possible to create a custom conversion file (MECH_CONV_FILE) and input this file to **ICON** by setting the ModMech variable to “user_defined” at compilation.

The horizontal grid and vertical layer structures for **ICON** are defined at execution through the input of a grid description (GRIDDESC) file and a meteorology cross-point 3-d (MET_CRO_3D) file, respectively. **ICON** interpolates between the input vertical layer structure and output layer structure if they are different.

5.5.2.1. ICON input files

Table 5.7. ICON input files

File Name	Format	Description
IC_PROFILE	ASCII	Vertical chemical profiles from which to derive initial conditions; this file is created by the user
MECH_CONV_FILE	ASCII	Mapping factors for converting between chemical mechanisms; this file is created by the user
CTM_CONC_1	GRDDED3	Name and location of the CMAQ concentration file from which to derive initial conditions; this file is output from the CCTM
CTM_PING_1	GRDDED3	Name and location of the plume-in-grid concentration file for the coarse domain; this file is output from the CCTM; only used when ICON_PING is set to “Y”
PING_PDM_1	GRDDED3	Name and location of the plume-in-grid plume dynamics file for the coarse domain; this file output from the PDM; only used when ICON_PING is set to “Y”
MET_CRO_3D_CRS	GRDDED3	Name and location of the course grid MET_CRO_3D file that is required for creating either plume-in-grid initial conditions or if the vertical grid structure changes between nested simulations; this file is output by MCIP
MET_CRO_3D_FIN	GRDDED3	Name and location of the fine grid MET_CRO_3D file that is required if the vertical grid structure changes between nested simulations; this file is output by MCIP
GRIDDESC	ASCII	Horizontal grid description file for defining the model grid; this file is output by MCIP or can be created by the user
MET_CRO_3D	GRDDED3	3-D cross point meteorology file for defining the vertical layer structure of the model grid; this file is output by MCIP

5.5.2.2. ICON compilation options

The configuration options listed here are set during compilation of the **ICON** executable. When these options are invoked they create a binary executable that is fixed to the specified configuration. To change these options it is necessary to recompile **ICON** and create a new executable.

- Opt:[default: verbose]
  Defines the action to be taken by the program **M3BLD** when extracting source code from CVS and compiling an executable.
  - *compile_all*: force compile, even if all the object files are current
  - *clean_up*: remove all source files upon successful compilation
  - *no_compile*: do everything except compile
  - *no_link*: do everything except link
  - *one_step*: compile and link in one step
  - *parse_only*: checks configuration file syntax
  - *show_only*: shows requested commands but does not execute them
  - *verbose*: shows requested commands as they are executed
- MakeOpt:
  Uncomment to build a Makefile to compile the executable
- ModInpt:
  Defines the format of the boundary conditions input files to be used by BCON.
  - *m3conc*: input a CCTM CONC file; used for nested simulations or windows of a parent domain
  - *profile*: input an ASCII vertical profiles file
- ModMech:
  Defines if the input boundary conditions data need to be converted from one chemical mechanism to another.
  - *mc_noop*: do not perform any mechanism conversion; used when extracting boundary conditions from a CCTM CONC file for a nested simulation or when the input profiles are already formatted for the correct mechanism
  - *user_defined*: input the file defined by the MECH_CONV_FILE variable; used for custom mechanism conversions
  - *radm2_to_cb4*: convert the input boundary conditions profiles from RADM2 to CB-IV speciation
  - *radm2_to_saprc99*: convert the input boundary conditions profiles from RADM2 to SAPRC99 speciation
- Mechanism:
  Specifies the gas-phase, aerosol, and aqueous phase chemical mechanisms to create boundary conditions for. The choices for the Mechanism variable are the mechanism directory names under the $M3MODEL/include/release directory. Examples include:
  - *cb4_ae3_aq*: CB-IV gas-phase mechanism, 3rd generation CMAQ aerosol mechanism, aqueous/cloud chemistry
  - *cb4_ae4_aq*: CB-IV gas-phase mechanism, 4th generation CMAQ aerosol mechanism, aqueous/cloud chemistry

- *saprc99_ae4_aq*: SAPRC99 gas phase mechanism, 4th generation CMAQ aerosol mechanism, aqueous/cloud chemistry
- *saprc99_ae3*: SAPRC99 gas phase mechanism and 3rd generation CMAQ aerosol mechanism

5.5.2.3. ICON compilation

Section 3.3 provides an overview of how to install and compile the CMAQ programs for the tutorial simulation. Follow the steps outlined in Section 3.3 to compile new versions of **ICON**:

1. If you have not already done so, build the CMAQ source code and compilation management program (**M3BLD**); this only needs to be done the first time you install CMAQ
2. Install and compile the I/O API and netCDF libraries; if these are already available from a previous CMAQ compilation, configure the **ICON** build script to use the available libraries
3. If you have not already done so, compile the STENEX library
4. Configure the **ICON** build script for your application
5. Invoke the build script to create an executable:

   ./bldit.icon.pgf

5.5.2.4. ICON execution options

The environment variables listed here are invoked during execution of the program and are set in the **ICON** run script.

- EXEC: [default: ICON_${CFG}]
  Executable to use for the simulation
- NPCOL_NPROW: [default: 1 1]
  Domain decomposition for parallel mode; ICON is normally run in a single processor environment, so this setting should always be “1 1”.
- GRIDDESC: [default: ../GRIDDESC1]
  Grid description file for setting the horizontal grid definition
- GRID_NAME:
  Name of the grid definition contained in the GRIDDESC file that specifies the horizontal grid for the current application of the model
- LAYER_FILE:
  Name and location of a MET_CRO_3D file for specifying the vertical layer structure for the current application of the model
- OUTDIR: [default: $M3DATA/icon]
  Output data directory
- IC:
  Sets the input file type. The setting of this variable determines how the input and output environment variables are set by the run script.

- *profile*: sets the output file name to include the tag “profile” in the name; uses the variable IC_PROFILE to point to an ASCII vertical profile file for input to **ICON**. Also optionally looks for the variable MECH_CONV_FILE to point to a user defined mechanism conversion file.

- *m3conc*: used for nested simulations, sets the output file name to include a start date in the name; uses the variable CTM_CONC_1 to point to a CCTM CONC file for input to **ICON**. Also looks for optional Plume-in-Grid inputs if the ICON_PING variable is set to “YES”.

- DATE:
  Sets the Julian date to use for tagging the ICON output file for nested runs.

- SDATE:
  Julian start date for extracting initial conditions from a CCTM CONC file for a nested simulation. If SDATE is not set, ICON will use the first hour of the CTM_CONC_1.

- STIME
  Start time for extracting boundary conditions from a CCTM CONC file for a nested simulation. If SDATE is not set, ICON will use the first hour of the CTM_CONC_1.

5.5.2.5. ICON output files

Table 5.8. ICON output files

File Name	Format	Description
INIT_CONC_1	GRDDED3	Gridded initial conditions data output on the model grid defined by GRID_NAME

The default location of the **ICON** output files is the $M3DATA/icon directory, controlled by the OUTDIR variable in the run script. The default naming convention for all **ICON** output files uses the APPL and GRID_NAME environment variables in the file name. For initial conditions created from existing CCTM CONC files, the Julian date is also used in the file name through the DATE environment variable. All of the file naming variables for **ICON** outputs are set in the run script.

5.6. JPROC

5.6.1. Description

JPROC calculates daily clear sky photolysis rates from look-up tables of molecular absorption cross-section and quantum yield data, and climatologically-derived ozone-column and optical depth data. The output from **JPROC** are ASCII look-up tables of daily clear sky photolysis rates for photochemical reactions in a selected gas-phase photochemical mechanism at different altitudes, latitudes, and hours from noon. The photochemical mechanism that these rates are derived for is selecting during compilation of **JPROC**. The altitudes (meters), latitudes (degrees), and hour angles (from noon) that the rates are derived for are hardwired in the JPROC source code.

5.6.2. Files, configuration, and environment variables

Figure 5.5. JPROC input and output files

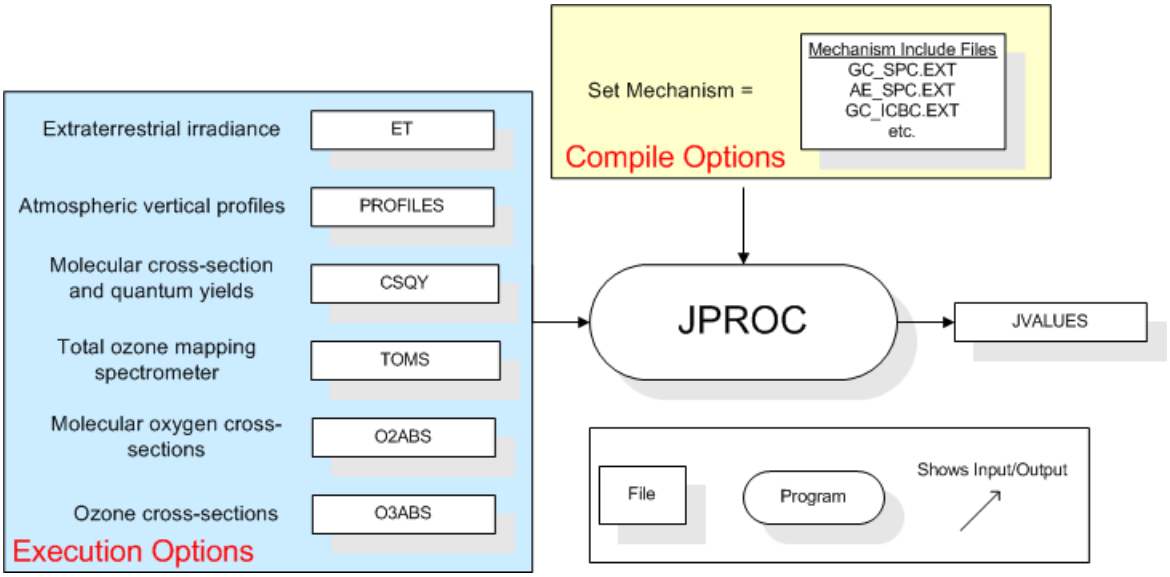


Figure 5 shows the input and output files for **JPROC**. Like the rest of the CMAQ programs, there is a distinction between the options that are invoked at compilation versus those invoked at execution of the program. When compiling **JPROC**, the user specifies a chemical mechanism to determine the gas-phase chemistry for which to calculate photolysis rates. Setting the Mechanism variable in the **JPROC** compile script configures the program to use a specific set of mechanism include files to build an executable. **JPROC** executables are hard-wired to a specific mechanism configuration.

While **JPROC** does not require any configuration at execution, there are several required and optional input files that the user must provide to the program. For the selected photochemical mechanism, the user must provide a set of molecular absorption cross-section and quantum yield data files (CSQY) that are consistent with the photolysis reactions in the mechanism. CMAQ is distributed with a full set of CSQY files for the CB-IV and SAPRC-99 mechanism versions supported by the model. If new mechanisms are added to CMAQ, the user must produce the appropriate CSQY data files for the added mechanism. The user also has the option of using the default atmospheric profiles contained in the PROFILES input file or using total ozone mapping spectrometer data (TOMS) to replace the climatologically-derived ozone column data in the PROFILES file.

5.6.2.1. JPROC input files

Table 5.9. JPROC input files

File Name	Format	Description
ET	ASCII	Extraterrestrial radiation as a function of wavelength
PROFILES	ASCII	Seasonal vertical profiles of ozone concentrations, aerosol attenuation, temperature, air pressure and Dobson values
TOMS	ASCII	Total ozone column measurements from the Total Ozone Mapping Spectrometer instrument aboard the sun-synchronous polar orbiting Nimbus satellite
O2ABS	ASCII	Absorption cross-section and quantum yield data for molecular oxygen as a function of wavelength
O3ABS	ASCII	Absorption cross-section and quantum yield data for ozone as a function of wavelength
CSQY	directory path	Directory path containing absorption cross-section and quantum yield data for gas-phase photolysis reactions as a function of wavelength

5.6.2.2. JPROC compilation options

The configuration options listed here are set during compilation of the **ICON** executable. When these options are invoked they create a binary executable that is fixed to the specified configuration. To change these options it is necessary to recompile **ICON** and create a new executable.

- Opt:[default: verbose]

  Defines the action to be taken by the program **M3BLD** when extracting source code from CVS and compiling an executable.

  - *compile_all*: force compile, even if all the object files are current
  - *clean_up*: remove all source files upon successful compilation
  - *no_compile*: do everything except compile
  - *no_link*: do everything except link
  - *one_step*: compile and link in one step
  - *parse_only*: checks configuration file syntax
  - *show_only*: shows requested commands but does not execute them
  - *verbose*: shows requested commands as they are executed

- MakeOpt:

  Uncomment to build a Makefile to compile the executable

- Mechanism:

  Specifies the gas-phase, aerosol, and aqueous phase chemical mechanisms to create initial conditions. The choices for the Mechanism variable are the mechanism directory names under the $M3MODEL/include/release directory. Examples include:

  - *cb4_ae3_aq*: CB-IV gas-phase mechanism, 3rd generation CMAQ aerosol mechanism, aqueous/cloud chemistry
  - *cb4_ae4_aq*: CB-IV gas-phase mechanism, 4th generation CMAQ aerosol mechanism, aqueous/cloud chemistry
  - *saprc99_ae4_aq*: SAPRC99 gas phase mechanism, 4th generation CMAQ aerosol mechanism, aqueous/cloud chemistry
  - *saprc99_ae3*: SAPRC99 gas phase mechanism and 3rd generation CMAQ aerosol mechanism

5.6.2.3. JPROC compilation

Section 3.3 provides an overview of how to install and compile the CMAQ programs for the tutorial simulation. Follow the steps outlined in Section 3.3 to compile new versions of **JPROC**:

1. If you have not already done so, build the CMAQ source code and compilation management program (**M3BLD**); this only needs to be done the first time you install CMAQ
2. Install and compile the I/O API and netCDF libraries; if these are already available from a previous CMAQ compilation, configure the **JPROC** build script to use the available libraries
3. Configure the **JPROC** build script for your application
4. Invoke the build script to create an executable:

   ```
   ./bldit.jproc.pgf
   ```

5.6.2.4. JPROC execution options

The environment variables listed here are invoked during execution of the program and are set in the **JPROC** run script.

- EXEC: [default: JPROC_${CFG}]
  Executable to use for the simulation

5.6.2.5. JPROC output files

Table 5.10. JPROC output files

File Name	Format	Description
JVfile (JTABLE_$Date}	ASCII	Daily clear-sky photolysis rates file

The default location of the **JPROC** output files is the $M3DATA/jproc directory, controlled by the OUTDIR variable in the run script. The default naming convention for all **JPROC** output files uses the Date environment variable in the file name, which is aliased to the STDATE environment variable in the run script.

5.7. MCIP

5.7.1. Description

The Meteorology-Chemistry Interface Processor (**MCIP**) converts meteorology data output from either the PSU/NCAR Mesoscale Model (MM5) or the Weather Research Forecast (WRF) model to I/O API formatted files that are compatible with CMAQ and the Sparse Matrix Operator Kernel Emissions (SMOKE) model. **MCIP** automatically determines if an input file is MM5 or WRF by trying to open the file as a netCDF file. If the file can be read as netCDF, **MCIP** assumes the input is a WRF dataset; otherwise, MM5 is assumed.

While most of the variables calculated in the meteorology model are simply passed through by **MCIP** to the output I/O API files, radiation fields and dry deposition velocities are calculated by **MCIP**. In addition, the user has the option of passing through the planetary boundary layer (PBL) heights calculated by the meteorology model (recommended) or recalculating the PBL with **MCIP**. Environment variables in the **MCIP** run script control these configuration settings.

MCIP has the capability to extract both temporal and spatial subsets of the input meteorology files.

- The run script allows the user to specify the beginning and end dates/times of the **MCIP** simulation; these dates/times can fall anywhere within the range of the input meteorology time period.
- Two types of horizontal domain windowing are allowed with **MCIP**. The boundary trim option uniformly trims grid cells off each of the four horizontal boundaries of the input meteorology grid. The non-uniform trim option specifies an offset from the lower left corner of the input meteorology domain and the number of cells in the X and Y directions from the revised origin to extract from the input domain. More information about how to invoke these options is provided in Section 5.7.2.4: **MCIP** execution options
- **MCIP** also provides the capability to reconfigure the vertical layer structure in the input meteorology through interpolation from the input structure to an output structure defined through sigma coordinates in the run script. Commonly referred to as layer collapsing, this option should be exercised with caution as it can significantly impact the conservation of energy assumption inherent in the meteorology through its affects to the predicted wind fields.

5.7.2. Files, configuration, and environment variables

Figure 5.6. MCIP input and output files

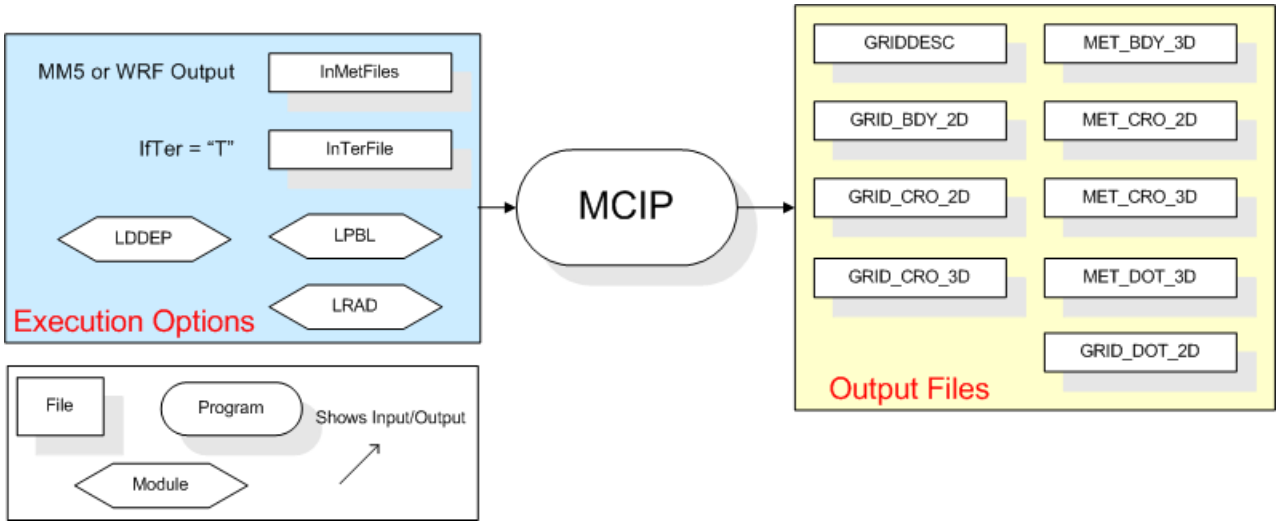


Figure 6 shows the input and output files and configuration options for **MCIP**. All **MCIP** configurations are accomplished at execution, a distinction from the rest of the CMAQ programs, which contain configuration options at both compilation and execution.

5.7.2.1. MCIP input files

Table 5.11. MCIP input files

File Name	Format	Description
InMetFiles	binary	List of MM5 or WRF output files for input to MCIP
InTerFile	binary	MM5 Terrain file with fractional land-use categories; used for calculating land-use dependent vertical diffusivity

5.7.2.2. MCIP compilation options

All model configuration options for **MCIP** are set during execution. System compiler options must be set in the provided Unix Makefile to build the program for different operating system/compiler combinations. Example compiler paths, flags, and library locations are provided in the default Makefile.

5.7.2.3. MCIP compilation

Section 3.3 provides an overview of how to install and compile the CMAQ programs for the tutorial simulation. Follow the steps outlined in Section 3.3 to compile new versions of **MCIP**:

1. Configure the Makefile for your operating system/compiler combination. Comment out the configuration that does not apply to your system. Uncomment the configuration that is closest to that of your system and make the necessary changes to point to the compiler path, I/O API location, and netCDF locations on your system.

2. Install and compile the I/O API and netCDF libraries if these are not already available on your system

3. Invoke the Makefile to create an executable by typing the following command in the directory that contains the Makefile and MCIP source code:
   ./make

5.7.2.4. MCIP execution options

The environment variables listed here are invoked during execution of the program and are set in the **MCIP** run script.

- APPL:
  Application name; scenario ID for file naming

- CoordName:
  Coordinate system name of the MCIP output grid that is written to the GRIDDESC file

- GridName:
  Model grid name of the MCIP output grid that is written to the GRIDDESC file

- DataPath:
  Input/output data directory path

- InMetDir:
  Path of the input data directory containing the MM5 or WRF output data files

- InTerDir:
  Path of the input data directory containing the MM5 TERRAIN file

- OutDir:
  Path of the **MCIP** output data directory

- ProgDir:[default: $cwd]
  Working directory containing the **MCIP** executable

- WorkDir:
  Temporary working directory for Fortran links and the namelist file

- InMetFiles:
  List of input meteorology files, including the directory path for each file; up to 100 files allowed as input to a single **MCIP** execution

- IfTer:[default: T]
  Binary flag indicating the availability of an input MM5 TERRAIN file; options include T (true) or F (false)

- InTerFile:
  Name and location of input MM5 TERRAIN file

- LPBL: [default: 1]
  Sets the source of the PBL values in the **MCIP** output files. The setting of this variable determines whether to use PBL values from the input meteorology file or to recalculate within **MCIP**.

  - *1*: use PBL values from the input meteorology

  - *2*: re-calculate PBL values within **MCIP** using PBL similarity

  - *3*: re-calculate PBL values within **MCIP** using surface layer similarity

- LRAD:[default: 1]
  Sets the source of the radiation fields in the **MCIP** output files. The setting of this variable determines whether to use the radiation fields from the input meteorology or to recalculate within **MCIP**.

  - *1*: use radiation fields from the input meteorology

  - *2*: re-calculate radiation fields using the **MCIP** version 1 algorithm

- LDDEP [default: 1]:
  Sets the dry deposition routine. The setting of this variable determines which dry deposition routine for calculating deposition velocities.

  - *1*: use the RADM (Wesely) dry deposition routine

  - *2*: use the Models-3 (Pleim) dry deposition routine

  - *1*: Use the Models-3 dry deposition routine and include Cl

  - *2*: Use the Models-3 dry deposition routine and include Cl and Hg

- MCIP_START:[format: YYYY-MM-DD-HH:MM:SS.SSSS]
  Beginning date and time [UTC] of data to output from **MCIP**. The start data and time must be contained within the input data from MM5 or WRF.

- MCIP_END:[format: YYYY-MM-DD-HH:MM:SS.SSSS]

End date and time [UTC] of data to output from **MCIP**. The end data and time must be contained within the input data from MM5 or WRF.

- INTVL: [default: 60]
  Output interval in minutes. This setting determines the amount of model time contained in each output time step.

- CTMLAYS:
  Sigma values of the vertical layers in the 3-d **MCIP** output. Comma delimited values for each sigma value must be in descending order starting at 1 and ending with 0. There are a maximum of 100 layers allowed. To use the all of the layers from the input meteorology without collapsing (or explicitly specifying), set CTMLAYS = -1.0.

- MKGRID: [default: T]
  Determines whether to output static (GRID) meteorology files

- • BTRIM: [default: 0]
  The number of boundary points to remove on each of the four horizontal sides of the **MCIP** domain. This setting affects the output **MCIP** horizontal domain by reducing the input meteorology domain by 2*BTRIM + 2*NTHIK + 1, where NTHIK is the lateral boundary thickness (from the BDY files). The extra point reflects the conversion from the grid points (dot points) to grid cells (cross points). Setting BTRIM = 0 will the maximum of the input meteorology domain. To remove the MM5 lateral boundaries, set BTRIM = 5.

  For windowing a subset domain of the input meteorology, set BTRIM = -1; this setting causes BTRIM to be replaced by the information provided by X0, Y0, NCOLS, and NROWS (see below).

- X0:
  X-coordinate of the lower-left corner of the full MCIP “X” domain (including the MCIP lateral boundary) based on the input MM5 domain. X0 refers to the east-west direction. This setting is only used when doing a subset of the input meteorology domain by setting BTRIM = -1.

- Y0:
  Determines whether to output static (GRID) meteorology files

- MKGRID: [default: T]
  Y-coordinate of the lower-left corner of the full MCIP “X” domain (including the MCIP lateral boundary) based on the input MM5 domain. Y0 refers to the north-south direction. This setting is only used when doing a subset of the input meteorology domain by setting BTRIM = -1.

- NCOLS:
  Number of columns in the output MCIP domain (excluding MCIP lateral boundaries)

- NROWS:
  Number of rows in the output MCIP domain (excluding MCIP lateral boundaries)

- LPRT_COL: [default: 0]
  Column cell coordinate for diagnostic outputs on the **MCIP** modeling domain

- LPRT_ROW: [default: 0]
  Row cell coordinate for diagnostic outputs on the **MCIP** modeling domain

5.7.2.5. MCIP output files

Table 5.12. MCIP output files

File Name	Format	Description
GRIDDESC	ASCII	Grid description file with coordinate and grid definition information
GRID_BDY_2D	BNDARY3	Time-independent 2-D boundary meteorology file
GRID_CRO_2D	GRDDED3	Time-independent 2-D cross point meteorology file
GRID_CRO_3D	GRDDED3	Time-independent 3-D cross point meteorology file
GRID_DOT_2D	GRDDED3	Time-independent 2-D dot point meteorology file
MET_BDY_3D	BNDARY3	Time-dependent 3-D boundary meteorology file
MET_CRO_2D	GRDDED3	Time-dependent 2-D cross point meteorology file
MET_CRO_3D	GRDDED3	Time-dependent 3-D cross point meteorology file
MET_DOT_3D	GRDDED3	Time-dependent 3-D dot point meteorology file

The default location of the **MCIP** output files is the $M3DATA/mcip3/$GridName directory. Since the default file names do not have any information about the model grid that they are simulating, the name of the grid is wired in the output directory path. The default naming convention for all **MCIP** output files uses only the APPL environment variable in the file name. All of the file naming variables for the **MCIP** outputs are set in the run script.

5.8. PARIO

5.8.1. Description

The **PARIO** library contains modules for controlling the model input and output in parallel multi-processor computing environments. In addition to a series of CMAQ-specific routines, it contains special implementations of several of the I/O API modules for multi-processor computing environments.

The **PARIO** library is only necessary when compiling the **CCTM** for parallel multi-processor environments; single processor versions of the **CCTM** and the other CMAQ programs do not use the **PARIO** library.

5.8.2. Files, configuration, and environment variables

5.8.2.1. PARIO input files

PARIO does not require any input files.

5.8.2.2. PARIO compilation options

Other than configuring the build script for your system (i.e. compiler and library locations) **PARIO** does not require any configuration at compilation.

5.8.2.3. PARIO compilation

Section 3.3 provides an overview of how to install and compile the CMAQ programs for the tutorial simulation. Follow the steps outlined in Section 3.3 to compile **PARIO**:

1. If you have not already done so, build the CMAQ source code and compilation management program (**M3BLD**); this only needs to be done the first time you install CMAQ

2. Install and compile the I/O API and MPICH libraries; if these are already available from a previous CMAQ compilation, configure the **PARIO** build script to use the available libraries

3. Invoke the build script to create an executable:

   ./bldit.pario.pgf

5.8.2.4. PARIO execution options

As **PARIO** is not a program it does not require any configuration at execution.

5.8.2.5. PARIO output files

Successful compilation of **PARIO** will produce the library file libpario.a. along with several module files in the $M3LIB/pario/$OS directory.

5.9. PDM

5.9.1. Description

The Plume Dynamics Model (**PDM**) is the plume in grid (PinG) preprocessor for PinG instrumented versions of the CCTM. Using input emissions files that identify the elevated point sources within the modeling domain to treat with PinG, the **PDM** simulates plume rise, plume transport, and plume dispersion for each of the selected point sources. The **PDM** also specifies when a particular plume cross section is to be transferred to the parent grid.

All **PDM** configuration happens during execution of the program. A series of environment variables for specifying different configurations of the **PDM** are located in the run script. By default these variables are commented out. To override the default setting of a configuration option, uncomment the corresponding variable and change the setting for your application.

5.9.2. Files, configuration, and environment variables

Figure 5.7. PDM input and output files

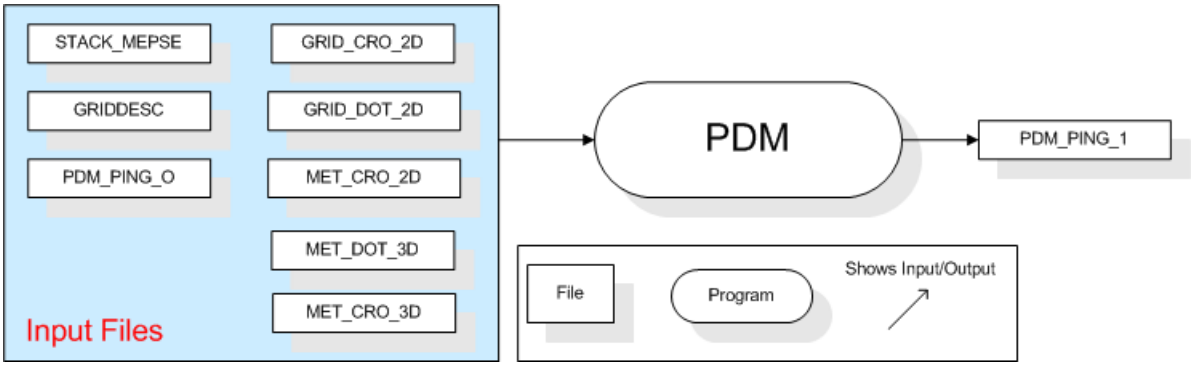


Figure 7 shows the input and output files for the PDM. The input file PDM_PING_O is only used when continuing a simulation from a previous day. Additional output files may also be written, in addition to the file shown in Figure 7. These diagnostic outputs are activated by setting the IPRFLG environment variable to “T” during execution.

5.9.2.1. PDM input files

Table 5.13. PDM input files

File Name	Format	Description
STACK_MEPSE	GRDDED3	GMajor emitting point source emissions file that identifies the elevated emissions sources to receive PinG treatment in the CCTM; output from and emissions program such as SMOKE or CONCEPT
GRID_CRO_2D	GRDDED3	Time-independent 2-D cross point meteorology file; this file is output by MCIP
GRID_DOT_2D	GRDDED3	Time-independent 2-D dot point meteorology file; this file is output by MCIP
MET_CRO_2D	GRDDED3	Time-dependent 2-D cross point meteorology file; this file is output by MCIP
MET_DOT_3D	GRDDED3	Time-dependent 3-D dot point meteorology file; this file is output by MCIP
MET_CRO_3D	GRDDED3	Time-dependent 3-D cross point meteorology file for defining the vertical layer structure of the model grid; this file is output by MCIP
PDM_PING_O	GRDDED3	Previous day PinG output file for restarting the simulation
GRIDDESC	ASCII	Horizontal grid description file for defining the model grid; this file is output by MCIP or can be created by the user

5.9.2.2. PDM compilation options

The configuration options listed here are set during compilation of the **PDM** executable. None of these options will affect the results of the **PDM** simulation; they only control the behavior of the program **M3BLD** in creating the **PDM** executable.

- Opt:[default: verbose]
  Defines the action to be taken by the program **M3BLD** when extracting source code from CVS and compiling an executable.
  - *compile_all*: force compile, even if all the object files are current
  - *clean_up*: remove all source files upon successful compilation
  - *no_compile*: do everything except compile
  - *no_link*: do everything except link
  - *one_step*: compile and link in one step

- *parse_only*: checks configuration file syntax
- *show_only*: shows requested commands but does not execute them
- *verbose*: shows requested commands as they are executed

- MakeOpt:
  Uncomment to build a Makefile to compile the executable

5.9.2.3. PDM compilation

Section 3.3 provides an overview of how to install and compile the CMAQ programs for the tutorial simulation. While the tutorial does not contain an exercise on using the **PDM**, the steps required to compile the **PDM** are virtually the same as those required to compile the other CMAQ programs, such as **BCON**. Follow these steps to compile the **PDM**:

1. If you have not already done so, build the CMAQ source code and compilation management program (**M3BLD**); this only needs to be done the first time you install CMAQ
2. Install and compile the I/O API and MPICH libraries; if these are already available from a previous CMAQ compilation, configure the **PDM** build script to use the available libraries
3. Configure the **PDM** build script for your system by setting the correction name and location of the Fortran compiler
4. Invoke the build script to create an executable:

   ```
   ./bldit.pdm.pgf
   ```

5.9.2.4. PDM execution options

- EXEC: [default: PDM_${CFG}]
  Executable to use for the simulation
- LOGFILE: [default: $BASE/$APPL.log]
  Uncomment to capture CCTM standard output to a log file; the LOGFILE variable sets the name and location of the log
- STDATE: [format: YYYYDDD]
  Beginning date of the **PDM** simulation
- STTIME: [format: HHMMSS]
  Beginning time of the **PDM** simulation
- NSTEPS: [format: HH]
  Duration of the **PDM** simulation
- IOLDFIL: [default: 0]
  Continuation flag; set to 0 for the first simulation day and 1 for all other days
- ICHUST: [default: 1]
  Methods for calculating the standard deviation of the turbulent component v and the Lagrangian time scale method
  - 1: Hanna et al. [1982]
  - 2: Hicks [1985] for convection boundary layer; Nieustadt [1984] for stable boundary layer; Arya [1984] for neutral boundary layer

- ICHUSY: [default: 2]
  Methods for calculating the standard deviation of the turbulent component y
  - 1: Irwin [1984]
  - 2: Weil [1988] and Venkatram [1988]
- ISHEAR: [default: 1]
  Wind shear effects on plume
  - 1: calculate wind shear effects
  - 0: omit wind shear effects
- IMETHD: [default: 1]
  Interpolation method [need more description]
  - 1: Linear
  - 0: PBL
- IDPLUM: [default: 0]
  Plume thickness calculation methods [need more description]
  - 0: Turner Method
  - 1: Temperature gradient method
- IPRFLG: [default: F]
  Flag for activating the creation of additional diagnostic files
  - F: omit extra diagnostic files
  - T: print extra diagnostic files
- IPARTFLG: [default: 0]
  Vertical partitioning flag [need more description]
- FACTC: [default: 1.0]
  Plume hand over criteria [need more description]
- INITC: [default: 1000]
  Plume initialization width (m)
- DDIRC: [default: 1.8]
  Cross-plume wind-shear factor [need more description]
- DSPDC: [default: 0.7]
  Wind speed shear factor [need more description]
- SPRFA: [default: 15]
  Vertical plume spread factor [need more description]
- SZOFA: [default: 3.545]
  Width factor for plume sections [need more description]
- HSIZE: [default: 1000]

Starting plume width size (m) [need more description]

- DISP: [default: keep]
  Controls the maintenance of existing log files.
  - delete: delete output log if it already exists
  - keep: abort simulation if output log exists
- OUTDIR: [default: $M3DATA/pdm]
  PDM output file directory location
- GRIDDESC: [default: ../GRIDDESC1]
  Grid description file for setting the horizontal grid definition
- GRID_NAME:
  Name of the grid definition contained in the GRIDDESC file that specifies the horizontal grid for the current application of the model
- LAYER_FILE:
  Name and location of a MET_CRO_3D file for specifying the vertical layer structure for the current application of the model

5.9.2.5. PDM output files

Table 5.14. PDM output files

File Name	Format	Description
PDM_PING_1	GRDDED3	Name and location of the time-dependent, 3-D active plume file
NFOUT_1		Optional diagnostic plume rise output
NFOUT_2		Optional diagnostic plume parameters
NFOUT_3		Standard diagnostic plume parameters

The default location of the **PDM** output files is the $M3DATA/pdm directory, controlled by the OUTDIR variable in the run script. The default naming convention for all **PDM** output files uses the APPL environment variable in the file name. All of the file naming variables for **PDM** outputs are set in the run script.

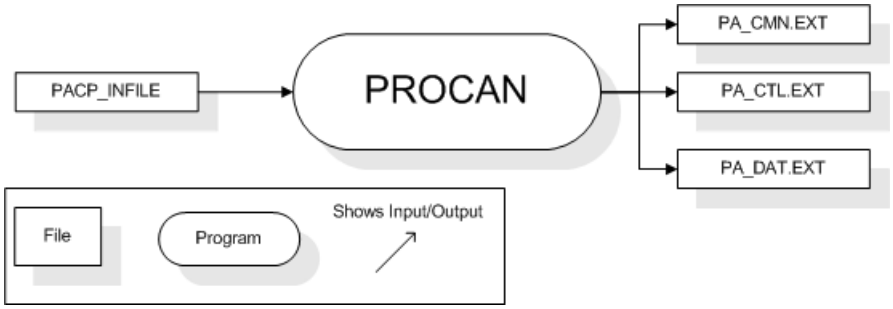
5.10. PROCAN

5.10.1. Description

must be followed to capture Process Analysis data rather than on what data should be captured and how it might be analyzed.

5.10.2. Files, configuration, and environment variables

Figure 5.8. PROCAN input and output files



The program PROCAN creates a set of include files needed to instrument the CCTM to produce process analysis output. This program reads and interprets instructions from a command file and then generates three Fortran include files (.EXT) used by CCTM to produce the Process Analysis outputs that were requested in the commands. The process analysis commands themselves are formatted according to a simple set of rules and a free-form format. Nevertheless, each command has a special syntax that must be followed, and each command makes use of special keywords and/or operators that have specific meaning to PROCAN. The commands are of three major types: global commands, IPR commands, and IRR commands. The discussion begins first, however, with a description of some general rules for configuring PROCAN.

5.10.2.1. PROCAN input files

Table 5.15. PROCAN input files

File Name	Format	Description
PACP_INFILE	ASCII	PROCAN command file specifies configuration options for the program

5.10.2.2. PROCAN configuration

PROCAN configuration is implemented through the command file PACP_INFILE. The free-form format of the PROCAN command file is similar to that used by the general mechanism processor, **CHEMECH**. In general, white spaces are ignored and line wrap is allowed (i.e., commands can be continued on a subsequent line after a hard return). The free-form format also allows embedded comments and makes use of special symbols to indicate the type of input data. Special rules for naming species, entering labels, and specifying numerical values, such as stoichiometric coefficients and rate constant parameters, are also used. Each major component of the command file is discussed below:

- *Comments*. All lines that have an exclamation point in column 1 are treated as comment lines and are ignored by PROCAN. Any text enclosed in braces ({ }) or parentheses (( )) is also treated as comment and ignored by PROCAN.

- *Species names*. PROCAN recognizes two types of species names: model species and user-defined Process Analysis species. “Model species” refer to species names in the Species Tables [Provide link to a table of species names for each mechanism]. These names must be spelled exactly as they appear in that table. The following special rules have been established for user-defined species names.

  - The Process Analysis species names must not contain any blanks and can be up to 16 characters long.

  - The name must begin with an alphabetic character but may contain any alphanumeric character (i.e., “A-Z”, “a-z”., and “0-9”) or the characters “:” and “_” after the first position.

  - The name is case sensitive. Thus, NO2 and no2 would represent two different species.
  - A name can have embedded comments but cannot span two lines.

- *Label names*. For some of the IRR commands, reaction labels appearing in the chemical mechanism reaction list input file can be referenced. These labels would normally be spelled exactly as they appear in the chemical mechanism reaction list input file, except embedded comments and their delimiters should be omitted. However, any embedded blanks in those label names should be omitted, and the label name should contain no more that 16 non-blank characters.
- *Numbers*. Numerical inputs in the command file can be either integer (e.g., 5), floating point (e.g., 5.0), or exponential (e.g., 5.0E+00). With the exponential format, the "E" may be either upper or lowercase; a positive exponent will be assumed if the sign of the exponent is missing.
- *Command Line Terminator*. Input command lines are terminated by a semicolon.

5.10.2.3. PROCAN commands

This section describes the individual process analysis commands that are used to construct a PROCAN command file. In the description of these commands, the following conventions will be used. Bold type is used for PROCAN keywords and normal type for user supplied inputs. Alternative inputs are separated by vertical bars (|), and optional inputs are enclosed in braces ({}). The PROCAN commands fall into three general categories:

- *Process Analysis Global Commands*. These commands include general specifications that are applicable throughout the configuration. These specific commands are described in Table 5-1.
- *Integrated Process Rate Command*he same number of time steps as the concentration output file. Since the I/O API currently has a limit of 120 output variables on a file, multiple files will be output if this limit is exceeded. These commands are described in Table 5-2.
- *Integrated Reaction Rate Commands*. The commands are specific to the configuration of the integrated reaction rates. The same considerations for file size and memory usage detailed above should be considered when using these commands. These commands are described in Table 5-4.

Table 5.16. Process Analysis Global Commands

Command	Description
OUTPUT_DOMAIN = {LOCOL[n_1] + HICOL[n_2] + LOROW[n_3] + HIROW[n_4] + LOLEV[n_5] + HILEV[n_6]};	The OUTPUT_DOMAIN command provides the capability to limit the IPR and IRR output data to a portion of the modeling domain. The ni in brackets are numbers that define the bounds of the output domain relative to the number of columns, rows, and vertical levels in the modeling domain. Thus, for example, the value for n1 must be greater than or equal to one and less than or equal to the number of columns in the domain. If any one domain specifier is omitted, the corresponding end of the modeling domain is used as a default. If the command is omitted entirely, output is generated for the entire domain.
DEFINE FAMILY familyname = {c_1*}species$_1$ {+ {c_2*}species$_2$ + ...};	The DEFINE FAMILY command is used to define a group of species as members of a family. The user specified "familyname" must be unique, and can be referenced in subsequent commands. The ci are numerical coefficients that default to one if not specified; "speciesi" represents the model species names.
ENDPA;	The ENDPA command signifies the end of the command input in the PACP command file.

Table 5.17. Integrated Process Rate Output Command

Command	Description
IPR_OUTPUT species|familyname|**ALL** = {pcode$_1$ + pcode$_2$ + ...};	The IPR_OUTPUT command defines specific IPR outputs to be generated during a CMAQ CTM simulation. A model species name, family name, or the keyword ALL must follow the IRR_OUTPUT keyword. The keyword ALL refers to all model species. IPRs are generated for the selected species or family, and they are controlled by the specified values of pcodei, where pcodei corresponds to one of the process codes listed below. If no process codes are specified, IPRs will be generated for every science process (i.e., the first 12 codes shown in Table 5-3). The output variables that are generated are named either species_pcodei or familyname_pcodei.

Table 5.18. IPR process codes

Process Code	Definition
XADV	Advection in the E-W direction
YADV	Advection in the N-S direction
ZADV	Vertical advection
ADJC	Mass adjustment
HDIF	Horizontal diffusion
VDIF	Vertical diffusion
EMIS	Emissions
DDEP	Dry deposition
CHEM	Chemistry
AERO	Aerosols
CLDS	Cloud processes and aqueous chemistry
PING	Plume-in-grid
XYADV	Sum of XADV and YADV
XYZADV	Sum of XADV, YADV, ZADV
TOTADV	Sum of XADV, YADV, ZADV, and ADJC
TOTDIF	Sum of HDIF and VDIF
TOTTRAN	Sum of XADV, YADV, ZADV, ADJC, HDIF, and VDIF

Table 5.19. Integrated Reaction Rate Output Commands

Command	Description
IRR_TYPE = FULL|PARTIAL|NONE;	The IRR_TYPE command defines the type of IRR analysis. With the type set to FULL, IRRs for each individual reaction will be calculated and written to the IRR output file, and all other IRR commands will be ignored. IRR_TYPE set to PARTIAL indicates that the IRR commands following this command are to be processed to produce user-defined IRR outputs. Type set to NONE causes all IRR commands to be ignored and no IRR output to be generated. If the command is omitted, type PARTIAL is assumed.
DEFINE RXNSUM cyclename = species$_1$;	The DEFINE CYCLE command is used to compute the net of all chemical production and loss of a species. Thus, this quantity is computed by summing the IRRs for all reactions in which a species is consumed, and then subtracting that sum from the sum of the IRRs for all reactions in which the species is produced. The “cyclename” is a user-defined name that must be unique, and can be referenced in subsequent IRR_OUTPUT commands.
DEFINE CYCLE sumname = {c_1*}<rxlabl$_1$> { ± {c_2*} <rxlabl$_2$> ± ...};	The RXSUM command is used to compute a linear combination of the IRRs for individual reactions that can then be referenced in a subsequent IRR_OUTPUT command; “sumname” is user-defined and must be unique. The linear combination of IRRs is defined according to the expressions following the equal signs that specify which reactions's IRRs to sum. The “rxlabli” is the reaction label that is used in the generalized mechanism. The “ci” are optional numerical coefficients that default to one if not specified.
IRR_OUTPUT irrname = {c_1*}op$_1$|cyclname{qual$_1$}| sumname{qual$_1$}| <rxlabl$_1$>{ ± {c_2*}op$_2$ | cyclname{qual$_2$}| sumname{qual$_2$}| <rxlabl$_2$> + ...};	The IRR_OUTPUT command defines a specific IRR output to be generated during a CMAQ simulation. It is constructed by specifying a linear combination of IRR operators, IRR global definitions, or IRRs for specified reactions. Each individual term in the combination must include either one of the five IRR operators described in Table M-14 (i.e., opi), a cycle name, a reaction sum name, or a reaction label enclosed in “greater than” and “less than” signs. The optional qualifiers (quali) for cyclename or reaction sum name can be either POSONLY or NEGONLY. With these qualifiers, the defined quantity is included as a term only when it is positive or negative, respectively. If the name is not qualified, the quantity is included regardless of sign. The numerical coefficients for each term (ci) are assumed to be one unless they are explicitly included. The irrname that is supplied by the user will be assigned as the variable name in the I/O API IRR output file.

Command	Description
DESCRIPTION = “description”;	The description command is provided to allow the user to specify a long description of the output variable that will be included on the I/O API IRR output name. If a description is not specified for an IRR_OUTPUT variable, the irrname (or short name) will be used in the output file. If the description command is used, it should be located immediately following the IRR_OUTPUT command to which it applies.
PROD [species$_1$] {**FROM**[species$_2$] {**AND**|**OR** [species$_3$] }}	The PROD operator is used to compute the total production of a species by summing the IRRs of all reactions in which species1 appears as a product. The optional qualifiers FROM and/OR restrict the sum to include only those reactions in which species2 and/or species3 are reactants. The “speciesi” can be any gas-phase mechanism species or a family of gas-phase species; “species2” or “species3” may also be the keyword HV to restrict the selection to photolytic reactions.
NETP [species$_1$] {**FROM**[species$_2$] {**AND**|**OR** [species$_3$] }}	The NETP operator is very similar to the production operator since it is used compute the production of a species. Whereas the PROD operator includes every reaction in which species1 occurs as a product, the NETP operator includes only those reactions in which the net production of species1 is greater than zero. Thus, if species1 or any member of the family species appears as both a reactant and a product with equal stoichiometry in a reaction, the PROD operator will include it, but the NETP operator will not. This operator is useful for getting the net production of a family, for example.
LOSS[species$_1$] {**AND**|**OR** [species$_2$] }	The LOSS operator is used to compute the total loss of a species by summing the IRRs of all reactions in which species1 appears as a reactant. The optional qualifier AND restricts the sum to include only those reactions in which both species1 and species2 are reactants. Similarly, the OR qualifier includes all reactions in which either “species1” or “species2” appears as a reactant, where “species1” or “species2” can be any gas-phase species in the mechanism, a family name that includes only gas-phase mechanism species, or the keyword HV to restrict the selection of reactions to those that are photolytic.
NETL[species$_1$] {**AND**|**OR** [species$_2$] }	The NETL operator is very similar to the loss operator since it is used to compute the loss of a species. However, it includes only those reactions in which there is a net loss of “species1” and/or “species2”. Thus, if species1 or any member of the family species appears as both a reactant and a product with equal stoichiometry in reaction, the NETL operator will not include it in summing the loss of that species, whereas the LOSS operator will include the IRR for that reaction.

Command	Description
NET[species$_1$]	The NET operator is similar to the CYCLE definition since it gives the net of the production and the loss of a species for all reactions in which “species1” appears either as reactant or a product; “species1” may be any gas-phase, mechanism species or any family consisting wholly of gas-phase mechanism species.

5.10.2.4. PROCAN execution variables

The environment variables listed here are invoked during execution of the program and are set in the PROCAN run script.

- EXEC: [default: PACP_${CFG}]
  Executable to use for the simulation
- PACP_INFILE: [default: $M3DATA/procan/pacp.inp]
  PROCAN control file for setting process analysis configuration

5.11. STENEX

5.11.1. Description

The Stencil Exchange (**STENEX**) library contains modules for controlling the communication between processors in parallel multi-processor computing environments. A noop version of the library is required for single processor versions of the **CCTM** and **ICON**; the parallel version is required when compiling for parallel multi-processor versions of the **CCTM**.

5.11.2. Files, configuration, and environment variables

5.11.2.1. STENEX input files

STENEX does not require any input files.

5.11.2.2. STENEX compilation options

Other than configuring the build script for your system (i.e. compiler and library locations) **STENEX** does not require any configuration at compilation.

5.11.2.3. STENEX compilation

Section 3.3 provides an overview of how to install and compile the CMAQ programs for the tutorial simulation. Follow the steps outlined in Section 3.3 to compile **STENEX**:

1. If you have not already done so, build the CMAQ source code and compilation management program (**M3BLD**); this only needs to be done the first time you install CMAQ
2. Install and compile the I/O API and MPICH libraries; if these are already available from a previous CMAQ compilation, configure the **STENEX** build script to use the available libraries
3. Invoke the single processor build script to create serial executables:

./bldit.se_noop.pgf

4. Invoke the multi-processor build script to create parallel executables:

   ./bldit.se.pgf

5.11.2.4. STENEX execution options

As **PARIO** is not a program it does not require any configuration at execution.

5.11.2.5. STENEX output files

Successful compilation of **STENEX** will produce the library files libsef90_noop.a for serial compilations and libse_snl.a, along with several module files, in the $M3LIB/stenex/$OS directory.

5.12. References

Arya, P.,1984: Parametric relations for the atmospheric boundary layer. *Boundary Layer Meteorology*, 30, 57-73.

Hanna, S.R., G.A. Briggs, and R.P. Hosker, 1982: Handbook on atmospheric diffusion, *U.S. DOE, DOE/TIC-11223, DE82002045, Nationa*

Hicks, BB. 1985: Behavior of turbulence statistics in the convective boundary layer, *J. of Climate and Applied Meteorology*. 24, 607-614.

Irwin, J.S. 1979: Scheme for estimating dispersion parameters as a function of release height, *EPA-600/4-79-062*, Research Triangle Park, N

Niewstadt, F.T.M., 1984: Some aspects of the turbulent stable boundary layer. *Boundary Layer Meteorology*, 30, 31-55.

Venkatram, A., 1988: Dispersion in the stable boundary layer. *Chapter 5, in Lectures on Air Pollution Modeling, A. Venkatram and J. Wynga*

Weil. J.C., 1988: Dispersion in the convective boundary layer. *Chapter 4, in Lectures on Air Pollution Modeling, A. Venkatram and J. Wynga*

Chapter 6. CMAQ FILES

Table of Contents

6.1. CMAQ Input Files 97
6.2. Basic CMAQ Output Files 131
6.3. Diagnostic and advanced CMAQ Output Files 133

The current version of CMAQ utilizes 21 input files for all of the preprocessors and the CCTM combined. The input files consist of a domain definition file for all programs; two file options for both ICON and BCON; one type of input file to MCIP; five mandatory and one optional input to JPROC; and for the CCTM an emissions, initial, and boundary conditions files, six files which define the meteorological conditions to be simulated, and a photolysis rates file. For many of these CCTM input file types, a separate data set is required for each grid mesh to be modeled.

CMAQ output files include a basic set of files with aerosol and gas-phase species concentrations, wet and dry deposition estimates, and visibility metrics, and an auxiliary set of output files for diagnosing model performance and plume-in-grid modeling.

6.1. CMAQ Input Files

This section describes each of the input files required by the different CMAQ programs. The section begins with a description of the grid definition file which is used by several CMAQ programs and then goes through a program-by-program listing of the CMAQ input file requirements. Table 4.1 lists the source, file type, and temporal and spatial dimensions of each CMAQ input file. Sample disk space requirements for desired input data set can be easily calculated from the information in Table 4.1; each data record is four bytes. Thus, if you know the number of variables in a CMAQ file and the spatial and temporal coverage of the data, the file sizes can be calculated.

Table 6.1. CMAQ Input File Information

File Name	File Type	Time-Dependence	Spatial Dimensions	Source
General				
GRIDDESC	ASCII	n/a	n/a	user/MCIP
ICON				
ic_profile	ASCII	Annual	n/a	user
m3conc	GRDDED3	Hourly	X * Y * Z	CCTM
BCON				
bc_profile	ASCII	Annual	n/a	user
m3conc	GRDDED3	Hourly	X * Y * Z	CCTM
JPROC				
Extraterrestrial Irradiance	ASCII	Annual	n/a	user
Default Atmospheric Profiles	ASCII	Annual	n/a	user
O_2 Absorption	ASCII	Annual	n/a	user
O_3 Absorption	ASCII	Annual	n/a	user
TOMS	ASCII	varies	n/a	user
CSQY	ASCII	Annual	n/a	user
MCIP				
MMOUT	Binary	Hourly	n/a	MM5
CCTM				
Initial Conditions	GRDDED3	Time-invariant	X * Y * Z	ICON/CCTM
Boundary Conditions	BNDARY3	Hourly	PERIM *Z	BCON
JTABLE	ASCII	Daily	n/a	JPROC
Emissions	GRDDED3	Hourly	X * Y * Z	SMOKE
2D Grid Cross	GRDDED3	Time-invariant	X * Y	MCIP
2D Grid Dot	GRDDED3	Time-invariant	(X+1)*(Y+1)	MCIP
3D Meteorological Boundary	BNDARY3	Hourly	PERIM*Z	MCIP
2D Meteorological Cross	GRDDED3	Hourly	X * Y	MCIP
3D Meteorological Cross	GRDDED3	Hourly	X * Y * Z	MCIP
3D Meteorological Dot	GRDDED3	Hourly	(X+1) * (Y+1) * Z	MCIP

6.1.1. GRIDDESC: Horizontal Domain Definition

Used by: JPROC, ICON, BCON, CCTM

The CMAQ grid description file (GRIDDESC) is used by all programs except JPROC and MCIP to define the horizontal spatial grid of the modeling domain. GRIDDESC is an I/O API ASCII file that contains two sections, a horizontal

coordinate section and grid description section. GRIDDESC is the logical name for text files which store horizontal coordinate and grid descriptions, and which is read by the DSCGRID() and DSCOORD() utility routines. This file has two segments. Each segment has a 1-line header (which by convention provides titles for the columns in the data records), a sequence of data records, and a terminal record with name field blank (i.e. ' ').

The first segment is the coordinate-system-description segment (Table 4.2) and consists of text records giving coordinate-system name and descriptive parameters P_ALP, P_BET, P_GAM, XCENT, and YCENT. Note that the standard names for UTM coordinate systems are given in the form UTM_<*nn*>, where *nn* is the number for the UTM zone. Most of the Eastern US would use coordinate system UTM_17, for example.

The second segment is the grid-description segment (Table 4.3), and consists of text records giving grid name, related coordinate-system name and descriptive parameters XORIG, YORIG, XCELL, YCELL, NCOLS, NROWS, and NTHIK. In the usual Models-3 modeling situation, there will be both "dot-point" and "cross-point" grids present (these grids are topological duals in the sense that the vertices (corners) of one correspond to the cell-centers of the other.

Each data record in these files consists of two or three list-formatted lines (i.e., items are separated by either blanks or commas). Name fields are quoted strings, and appear on the first of these lines. Numeric fields are given in double precision, and occur on either the second line or the second and third (this allows you to organize the text so that it is easily viewed in a text editor without running off-screen). The records have the following organization, depending upon whether they are in the first or second segments:

```
    COORD-NAME
COORDTYPE, P_ALP, P_BET, P_GAM
XCENT, YCENT

    or

COORD-NAME
COORDTYPE, P_ALP, P_BET, P_GAM, XCENT, YCENT

    and

GRID-NAME
COORD-NAME, XORIG, YORIG, XCELL, YCELL
NCOLS, NROWS, NTHIK

    or

GRID-NAME
COORD-NAME, XORIG, YORIG, XCELL, YCELL, NCOLS, NROWS, NTHIK
```

There are at most 32 coordinate systems and 256 grids listed in one of these files. These files are small enough to be archived easily with a study, and have a sufficiently simple format that new ones can easily be constructed "by hand." The GRIDDESC file is a list-formatted file, delimited by blank spaces unless otherwise specified.

An example of the two sections of a GRIDDESC file follows:

```
' '
'LAM_40N100W'
 2 30.0 60.0 -100.0 -100.0 40.0
' '
```

```
'M_32_99TUT02'
'LAM_40N100W'  544000.0 -992000.0  32000.0  32000.0  38  38 1
' '
```

The first of the two sections in this example GRIDDESC entry defines a horizontal coordinate named "LAM_40N100W". The coordinate definition is for a Lambert Conformal Grid, keyed by first column of the coordinate description line, which corresponds to the numerical code for the different I/O API-supported grid types (2 = Lambert). The next two parameters are the Lambert Alpha and Beta latitudes that determine the projection cone. The fourth parameter is the central meridian of the projection. The last two parameters are the latitude-longitude coordinates of the grid center of the Cartesian coordinate system.

The example grid definition above describes a grid named "M_32_99TUT02". The definition of the grid begins with a reference to a coordinate name from the coordinate definition section of the file; in this example the coordinate named "LAM_40N100W" is referenced in the grid definition. The next two parameters in the grid definition are the East-West and North-South offsets from the origin in meters, respectively. The next two parameters are the grid cell size in meters for the X and Y directions, followed by the number of cells in the X and Y directions. The grid definition concludes with the number of boundary cells.

Table 6.2. Coordinate system description segment

Line	Column	Name	Type	Description
1	A	Header	String	Single quote delimited header describing section contents; may be blank, i.e. ' '
2	A	COORD-NAME	String	Name of the coordinate description (required); single quote delimited
3	A	COORDTYPE	Int	County Code (required)
	B	P_ALP	Double	First map projection descriptive parameter (dependent on projection type)
	C	P_BET	Double	Second map projection descriptive parameter (dependent on projection type)
	D	P_GAM	Double	Third map projection descriptive parameter (dependent on projection type)
	E	XCENT	Double	Longitude for coordinate system origin (where X=0)
	F	YCENT	Double	Longitude for coordinate system origin (where Y=0)

Table 6.3. Coordinate system description segment

Line	Column	Name	Type	Description
1	A	Header	String	Single quote delimited header describing section contents; may be blank, i.e. ' '
2	A	GRID-NAME	String	Name of the horizontal grid (required); single quote delimited
3	A	COORD-NAME	String	Reference to a coordinate description (required), single quote delimited
	B	XORIG	Double	X-coordinate for lower-left (southwest) corner of the grid (dependent on projection type)
	C	YORIG	Double	Y-coordinate for lower-left (southwest) corner of the grid (dependent on projection type)
	D	XCELL	Double	X-coordinate grid cell size (dependent on projection type)
	E	YCELL	Double	Y-coordinate grid cell size (dependent on projection type)
	F	NCOLS	Int	Number of horizontal grid columns (dependent on projection type)
	G	NROWS	Int	Number of horizontal grid rows (dependent on projection type)
	H	NTHIK	Int	Boundary perimeter thickness (optional)

The GRIDDESC file can either be created by hand or generated automatically with MCIP. Starting with MCIP version 2.3, the program outputs a GRIDDESC file that is compatible with the other CMAQ programs.

6.1.2. IC_PROFILE: Initial conditions vertical profiles

Used by: ICON

ICON can generate initial conditions from two different input file types. The first file type is an ASCII vertical profile file that list species concentrations at different model layers that are fixed in space in time. ICON is configured to generate initial conditions from ASCII vertical profiles by choosing the "prof" input module when compiling the programs (see Section xxx on ICON).

The ASCII-formatted vertical profile files begin with a header that contains a comment section describing the data and a file description section that defines the number of sigma levels in the file, the number of pollutants in the file, and the vertical coordinate values of the different sigma levels. The next entries in these files are the Julian start date and the start time of the data and are not used by ICON.

Each line in the IC_PROFILE file corresponds to a different pollutant and begins with the name of the pollutant. The subsequent columns on each line list the chemical concentration at each layer contained in the file. Gas-phase species are provided in ppmV units and aerosol species are provided in $\mu g/m^3$ units in the IC_PROFILE file. The layer structure of the ICON vertical profiles does not need to correspond exactly to the layer structure that will be modeled; the program will interpolate the data to the correct vertical format for the simulation.

Initial conditions are inherently temporally static, as they correspond to the first hour of a model simulation only. Generating initial conditions from an ASCII profile file produces a gridded file for input to the CCTM that has uniform species concentrations across the entire horizontal grid at each model layer. For spatially resolved initial conditions, it is necessary to use the other input file type to ICON, an existing CCTM concentration file (CTM_CONC_1).

A detailed description of the vertical profile file format for initial conditions is provided in Table 4.4. The header of the profiles file is list-formatted while the data section of the file is fixed format.

Table 6.4. IC_PROFILE format description

Line	Column	Name	Type	Description
1-3		Text Header	String	Text description of the contents and source of the initial conditions file (optional)
4	A	NUM_SIGMA_LVL	Int	Number of sigma levels contained in the file (required)
	B	NUM_POLL	Int	Number of pollutants contained in the file (required)
	C	SIGMA_LVL	Real	Vertical coordinate values of sigma-p levels, number of values (n) matches NUM_SIGMA_LVL (required)
4	n	...	...	...
5	A	STDATE	String	Julian start date of the file, YYYYDDD (optional)
	B	STTIME	String	Start time of the file, HH (optional)
6	1-10	SPECIES1	String	Pollutant name, enclosed in double quotes (required)
	12-20	LAYER1_IC	Exp	IC concentration for species 1 in lowest sigma layer (required)
	23-31	LAYER2_IC	Exp	IC concentration for species 1 in 2nd sigma layer (required)
	34-42	LAYER3_IC	Exp	IC concentration for species 1 in 3rd sigma layer (required)
	45-53	LAYER4_IC	Exp	IC concentration for species 1 in 4th sigma layer (required)
	...	LAYERX_IC	Exp	IC concentration for species 1 in Xth sigma layer (required)
7	1-10	SPECIES2	String	Pollutant name, enclosed in double quotes (required)

Line	Column	Name	Type	Description
	12-20	LAYER1_IC	Exp	IC concentration for species 2 in lowest sigma layer (required)
	23-31	LAYER2_IC	Exp	IC concentration for species 2 in 2nd sigma layer (required)
	34-42	LAYER3_IC	Exp	IC concentration for species 2 in 3rd sigma layer (required)
	45-53	LAYER4_IC	Exp	IC concentration for species 2 in 4th sigma layer (required)
	...	LAYERX_IC	Exp	IC concentration for species 2 in Xth sigma layer (required)
...	...	...	...	...
	...	LAYERX_IC	Exp	IC concentration for species 2 in Xth sigma layer (required)
...	...	...	...	...
Y	1-10	SPECIESY	String	Pollutant name, enclosed in double quotes (required)
...	12-20	LAYER1_IC	Exp	IC concentration for species Y in lowest sigma layer (required)
...	23-31	LAYER2_IC	Exp	IC concentration for species Y in 2nd sigma layer (required)
...	34-42	LAYER3_IC	Exp	IC concentration for species Y in 3rd sigma layer (required)
...	45-53	LAYER4_IC	Exp	IC concentration for species Y in 4th sigma layer (required)
...	...	LAYERX_IC	Exp	IC concentration for species Y in Xth sigma layer (required)

A sample of the four sections of an IC_PROFILE file is shown below.

Optional initial condition: The vertical coordinate of the model to generate these b.c. is the terrain-following sigma coordinate. The number of sigma layers and defined sigma levels are listed below.
6 55 1.00 0.98 0.93 0.84 0.60 0.30 0.00

```
 1988180 00
"SO2 " 0.300E-03 0.200E-03 0.100E-03 0.100E-03 0.200E-04 0.100E-04
```

6.1.3. BC_PROFILE: Boundary conditions vertical profiles

Used by: BCON

Like ICON, BCON can generate boundary conditions from two different input file types. The first file type is an ASCII vertical profile file that list species concentrations at different model layers that are fixed in space in time. BCON is configured to generate initial conditions from ASCII vertical profiles by choosing the "prof" input module when compiling the programs (see Section xxx on BCON).

The ASCII-formatted vertical profile files begin with a header that contains a comment section describing the data and a file description section that defines the number of sigma levels in the file, the number of pollutants in the file, and the vertical coordinate values of the different sigma levels. The next entries in these files are the Julian start date and the start time of the data and are not used by BCON. The only difference between the ICON and BCON vertical profile files is that the BCON input consists of four data sections, corresponding to each horizontal boundary (e.g. north, south, east, west) of the model grid. The BCON input profiles contain an additional field that precedes each data section indicating which horizontal boundary the data section describes.

The format of the data sections in the ICON and BCON input files are exactly the same. Each line corresponds to a different pollutant and begins with the name of the pollutant. The subsequent columns on each line list the chemical concentration at each layer contained in the file. Gas-phase species are provided in ppmV units and aerosol species are provided in µg/m3 units in the BCON inputs. The layer structure of the BCON vertical profiles does not need to correspond exactly to the layer structure that will be modeled; the programs will interpolate the data to the correct vertical format for the simulation.

Boundary conditions can either be time-independent (static) or time-dependent (dynamic). Boundary conditions generated with the ASCII vertical profiles are both static and spatially uniform along each of the four horizontal boundaries. For spatially and temporally resolved boundary conditions, it is necessary to use the other input file type to BCON, an existing CCTM concentration file (CTM_CONC_1).

A detailed description of the vertical profile file format for boundary conditions is provided in Table 4.5. The header of the profiles file is list-formatted while the data section of the file is fixed format.

Table 6.5. BC_PROFILE format description

Line	Column	Name	Type	Description
1-3		Text Header	String	Text description of the contents and source of the initial conditions file (optional)
4	A	NUM_SIGMA_LVL	Int	Number of sigma levels contained in the file (required)
	B	NUM_POLL	Int	Number of pollutants contained in the file (required)
	C	SIGMA_LVL	Real	Vertical coordinate values of sigma-p levels, number of values (n) matches NUM_SIGMA_LVL (required)
4	n	...	...	...
5	A	STDATE	String	Julian start date of the file, YYYYDDD (optional)
	B	STTIME	String	Start time of the file, HH (optional)
6	A	Direction	String	North, South, East, West; indicates the boundary described by the subsequent data section (required)
6	1-10	SPECIES1	String	Pollutant name, enclosed in double quotes (required)
	12-20	LAYER1_IC	Exp	IC concentration for species 1 in lowest sigma layer (required)
	23-31	LAYER2_IC	Exp	IC concentration for species 1 in 2nd sigma layer (required)
	34-42	LAYER3_IC	Exp	IC concentration for species 1 in 3rd sigma layer (required)
	45-53	LAYER4_IC	Exp	IC concentration for species 1 in 4th sigma layer (required)
	...	LAYERX_IC	Exp	IC concentration for species 1 in Xth sigma layer (required)

Line	Column	Name	Type	Description
7	1-10	SPECIES2	String	Pollutant name, enclosed in double quotes (required)
	12-20	LAYER1_IC	Exp	IC concentration for species 2 in lowest sigma layer (required)
	23-31	LAYER2_IC	Exp	IC concentration for species 2 in 2nd sigma layer (required)
	34-42	LAYER3_IC	Exp	IC concentration for species 2 in 3rd sigma layer (required)
	45-53	LAYER4_IC	Exp	IC concentration for species 2 in 4th sigma layer (required)
	...	LAYERX_IC	Exp	IC concentration for species 2 in Xth sigma layer (required)
...	...	...	...	...
Y	A	Direction	String	North, South, East, West indicates the horizontal boundary described by the subsequent data section (required)
Y+1	1-10	SPECIESY	String	Pollutant name, enclosed in double quotes (required)
...	12-20	LAYER1_IC	Exp	IC concentration for species Y in lowest sigma layer (required)
...	23-31	LAYER2_IC	Exp	IC concentration for species Y in 2nd sigma layer (required)
...	34-42	LAYER3_IC	Exp	IC concentration for species Y in 3rd sigma layer (required)
...	45-53	LAYER4_IC	Exp	IC concentration for species Y in 4th sigma layer (required)
...	...	LAYERX_IC	Exp	IC concentration for species Y in Xth sigma layer (required)

A sample of the important sections of a BC_PROFILE file is shown below.

```
 Optional boundary condition: The vertical coordinate of the model to generate
 these b.c. is the terrain-following sigma coordinate. The number of sigma
 layers and defined sigma levels are listed below.
   6 55 1.00 0.98 0.93 0.84 0.60 0.30 0.00
 1988180 00
North
"SO2 " 0.300E-03 0.200E-03 0.100E-03 0.100E-03 0.200E-04 0.100E-04
West
"SO2 " 0.300E-03 0.200E-03 0.100E-03 0.100E-03 0.200E-04 0.100E-04
```

6.1.4. MECH_CONV_FILE: Mechanism Conversion File

Used by: ICON, BCON

MECH_CONV_FILE is the logical name of the ASCII input file that contains photochemical mechanism conversion rules. This file contains expressions that are used to convert IC's and BC's from one mechanism form to another. The default IC and BC profiles distributed with CMAQ contain RADM2 species. The ICON and BCON processors have a hardwired routine for converting from RADM2 to CB-IV. This hardwired routine can be overridden by supplying a MECH_CONV_FILE at run time.

A detailed description of the MECH_CONV_FILE format is provided in Table 4.6. The file is list-formatted.

Table 6.6. MECH_CONV_FILE format description

Line	Column	Name	Type	Description
1	1-16	SPP1_OUT	String	Output mechanism species name (required)
	17	"="	Char	" " delimits the left-hand and right-hand side of the conversion equation (required)
	18	SPP1_IN	String	Input mechanism species name, coefficients on right-hand side of equation may be either integer, real, or exponential format (required)
1	n	...	...	...
1	n+1	";"	Char	";" conversion rule terminator, allows rules to span multiple lines (required)
2	1-16	SPP2_OUT	String	Output mechanism species name (required)
	17	"="	Char	"=" delimits the left-hand and right-hand side of the conversion equation (required)
	18	SPP2_IN	String	Input mechanism species name, coefficients on right-hand side of equation may be either integer, real, or exponential format (required)
2	n	...	...	...
2	n+1	";"	Char	";" conversion rule terminator, allows rules to span multiple lines (required)
...	...	...	...	...
X	1-16	SPPX_OUT	String	Output mechanism species name (required)
	17	"="	Char	"=" delimits the left-hand and right-hand side of the conversion equation (required)

Line	Column	Name	Type	Description
	18	SPPX_IN	String	Input mechanism species name, coefficients on right-hand side of equation may be either integer, real, or exponential format (required)
X	n	...	...	...
X	n+1	";"	Char	";" conversion rule terminator, allows rules to span multiple lines (required)
X+1	1	"END;"	String	"END;" file terminator (required)

A sample of the entries in a MECH_CONV_FILE file is shown below.

```
 NO2  =  NO2;
  FORM  = HCHO + 1.0 * GLY;
 END;
```

6.1.5. CTM_CONC_1: CCTM concentration Files

Used by: ICON, BCON

The second file type that can be input to ICON and BCON is a GRDDED3-formatted CCTM output concentration file. ICON and BCON are configured to generate initial and boundary conditions from a CCTM concentration file by selecting the "m3conc" option for the input file module when compiling the programs. The input concentration file must cover a temporal and spatial extent that is consistent with the time period and domain that is being modeled, respectively. Both ICON and BCON will require that a Julian start date be specified at execution that identifies the first time step to extract from the input concentration file; BCON also requires a run length specification to direct the number of time steps of boundary conditions to extract from the input file. For nested runs, the nested domain for which initial and boundary conditions are being extracted must be on the same projection and fall within the domain contained in the input concentration file.

6.1.6. CSQY: Absorption cross section and quantum yields

Used by: JPROC

CSQY is the logical name for the ASCII data file containing absorption cross-section and quantum yield data for unique photolysis reactions. The data in these files are listed as a function of wavelength and correspond to the standard photolysis reactions in each of the chemical mechanisms implemented in CMAQ. A flexible format allows users to deviate from these standards and test new data with a trivial amount of effort.

The ASCII-formatted CSQY files begin with a header describing the applicable photolysis reaction, comments about the source of the data, and the location on the wavelength interval that the data represents (Beginning, Centered, or Ending). Comment lines are preceded with a "!". The last section of the header is a multiplier (FAC) that is applied to the photolysis rate calculation. The data section of the CSQY file lists the wavelength of the incoming solar radiation (nm), the absorption cross section (cm) and the quantum yield as columns, with each row corresponding to a specific

wavelength interval. The CSQY file uses a space delimited, free-form format for the data section of the file. A detailed description of the CSQY file format is provided in Table 4.7.

Table 6.7. CSQY format description

Line	Column	Name	Type	Description
1	A	Reaction ID	String	Text name identifying the data; this name is cross-referenced in the chemical mechanism description and include files (required)
2	A	Comments	String	Preceded by "!" comment lines describe the reaction, list the stoichiomtery, and document the source of the data (optional)
n	...	...	...	...
n + 1	A	Data Location	String	Field indicating the location of the data as measured across the wavelength band; possible answers: beginning, ending, centered, point (required)
n + 2	A	Multiplier	String	Multiplication factor to apply to photolysis rate equation; line begins with FAC=, factor listed in real or exponential format (required)
n + 3	A	Wavelength	Int or Real	Wavelength corresponding to CSQY data, units in nm (required)
	B	Absorption Cross-Section	Real or Exp	Measurement of the cross-section of a molecules spherical receiving surface for actinic flux, units in cm2 molecule-1 (required)
	C	Quantum Yield	Real	Ratio of the number of molecules reacting via a specific pathway to the number of molecules absorbing photons in that wavelength interval, units in molecules photon-1 (required)
n + 4	A	Wavelength	Int	Wavelength corresponding to CSQY data, units in nm (required)

Line	Column	Name	Type	Description
	B	Absorption Cross-Section	Real or Exp	Measurement of the cross-section of a molecules spherical receiving surface for actinic flux, units in cm2 molecule-1 (required)
	C	Quantum Yield	Real	Ratio of the number of molecules reacting via a specific pathway to the number of molecules absorbing photons in that wavelength interval, units in molecules photon-1 (required)
n + X	...	...	...	...

A sample of the important sections of a CSQY file is shown in Table 4.8.

Table 6.8. Example CSQY data file

```
1       ALD_CBIV88
2       ! Acetaldehyde Photolysis (ALD)
3       ! CH3CHO + hv (+2O2)→CH3OO + HO2 + CO
4       ! Taken from Gery et al. [1988]; CSQY from Baulch et al. 5 [1984].
6       ! format:  wl, abs_cs, qy
7       Centered
8       ! With FAC, units are (cm²/molecule)
9       FAC=1.0E-20
10       280  4.50  0.580
11       281  4.54  0.575
12       282  4.58  0.570
```

6.1.7. ET: Extraterrestrial irradiance

Used by: JPROC

ET is the logical name for the ASCII data file containing extraterrestrial radiation as a function of wavelength. The extra-terrestrial irradiance file has a format similar to the CSQY file. The file begins with a header section; comment lines are preceded with a "!". Like the CSQY file, the header contains a field describing the location on the wavelength interval that the data represents and a multiplier. The data section uses a space delimited, free-form format and lists the wavelength of the incoming solar radiation (nm) and the irradiance (Photons/cm²-s) at each wavelength, with each row corresponding to a specific wavelength interval. A detailed description of the file format is provided in Table 4.9.

Table 6.9. ET file format description

Line	Column	Name	Type	Description
1	A	Comments	String	Preceded by "!" comment lines, describe the file contents and document the source of the data (optional)
n	...	...	...	...
n + 1	A	Data Location	String	Field indicating the location of the data as measured across the wavelength band; possible answers: beginning, ending, centered, point (required)
n + 2	A	Multiplier	String	Multiplication factor to apply to photolysis rate equation; line begins with FAC=, factor listed in real or exponential format (required)
n + 3	A	Wavelength	Int or Real	Wavelength corresponding to ET data, units in nm (required)
	B	Extraterrestrial Irradiance	Real or Exp	Estimation of the photon flux reaching the exterior of the earths atmosphere, units in photons cm^{-2} $second^{-1}$ (required)
n + 4	A	Wavelength	Int	Wavelength corresponding to ET data, units in nm (required)
	B	Extraterrestrial Irradiance	Real or Exp	Estimation of the photon flux reaching the exterior of the earths atmosphere, units in photons cm^{-2} $second^{-1}$ (required)
n + X	...	...	...	...

A sample of the important sections of an ET file is shown below.

```
! Extra Terrestrial Irradiance
! taken from the RADM data---derived from the WMO 1985 report Table 7-4
! format: wl, et_irrad
Beginning
! With FAC, units are (Photons cm-2 s-1)
```

FAC=1.0
185.185 3.620E+11
186.916 4.730E+11

6.1.8. PROFILES: Atmospheric vertical profiles

Used by: JPROC

PROFILES is the logical name for the ASCII data file containing seasonal and vertical profiles for ozone, aerosol attenuation, temperature, air pressure, and Dobson values. The ASCII-formatted data provided in the atmospheric profiles data file (PROFILES) are at 19 latitude bands (90°N to 90°S) and 51 altitudes (0 to 50 km) in three distinct data sections. The first data section contains seasonal, latitude-dependent vertical profiles of O_3 concentrations (molecules/cm³), air temperature (K), and air density (molecules/cm³). The second data section contains monthly Dobson values at the 19 latitude bands. The last data section contains vertical profiles from the 1976 U.S. Standard Atmosphere of air temperature (K), air density (molecules/cm³), ozone concentrations (molecules/cm³), and aerosol attenuation coefficients (km^{-1}).

The first data section of the PROFILES file is divided into 228 (19*3*4) data blocks with each block representing one of the three variables (O_3, air temperature, and air density) at one of the 19 latitude bands, for each of the 4 seasons of the year. The individual data blocks contain 51 values per variable, representing standard conditions at altitudes ranging from 0-50 km. The data are ordered, from general to specific, by season (spring, summer, autumn, winter), variable (O_3, air temperature, air density), latitude, and altitude. For example, the first block in the PROFILES file contains spring O_3 concentration profiles at the latitude band ranging from 90°N to 80°N from 0 to 50 km above sea level, the first value in the block is at 0 km and the last value is at 50 km. The next data block is again spring O_3 concentration profiles but at the latitude band ranging from 80°N to 70°N. The next 17 data blocks complete the spring O_3 concentration profiles by continuing through rest the 19 latitude bands with the last block representing the 80°S to 90°S latitude band. The 20th data block begins the spring air temperature profiles at the latitude band ranging from 90°N to 80°N and is followed by 18 more data blocks of spring air temperature profiles. The following 19 data blocks follow an identical format for air density and round out the spring profiles. These 19*3 data blocks are then repeated for summer profiles, autumn profiles, and finally winter profiles.

The second data section in the PROFILES file contains monthly average Dobson Units. The data are organized in twelve rows (January through December) and 19 columns (90°N to 80°N through 80°S to 90°S).

The last data section in the PROFILES file contains vertical profiles from the 1976 U.S. Standard Atmosphere of temperature (K), air density (molecules/cm³), ozone concentrations (molecules/cm³), and aerosol attenuation coefficients (km^{-1}). The data are organized in 51 rows, corresponding to altitude (0-50km), with four columns per row for each of the four variables mentioned above. A detailed description of the file format is provided in Table 4.10.

Table 6.10. PROFILE file format description

Line	Column	Name	Type	Description
1	A	Ozone concentration at Season 1, Latitude 1, Level 1	Exp (E10.3)	Ozone measurements as a function of season, latitude, and vertical level; units in molecules cm^{-3} (required)
	B	Ozone concentration at Season 1, Latitude 1, Level 2	Exp (E10.3)	Ozone measurements as a function of season, latitude, and vertical level; units in molecules cm^{-3} (required)
...	...	...	...	...
127	A	Ozone concentration at Season 1, Latitude 19, Level 1	Exp (E10.3)	Ozone measurements as a function of season, latitude, and vertical level; units in molecules cm^{-3} (required)
	B	Ozone concentration at Season 1, Latitude 19, Level 2	Exp (E10.3)	Ozone measurements as a function of season, latitude, and vertical level; units in molecules cm^{-3} (required)
...	...	...	...	...
134	A	Temperature profiles at Season 1, Latitude 1, Level 1	Exp (E10.3)	Temperature measurements as a function of season, latitude, and vertical level; units in K (required)
	B	Temperature profiles at Season 1, Latitude 1, Level 2	Exp (E10.3)	Temperature measurements as a function of season, latitude, and vertical level; units in K (required)
...	...	...	...	...
260	A	Temperature profiles at Season 1, Latitude 19, Level 1	Exp (E10.3)	Temperature measurements as a function of season, latitude, and vertical level; units in K (required)
	B	Temperature profiles at Season 1, Latitude 19, Level 2	Exp (E10.3)	Temperature measurements as a function of season, latitude, and vertical level; units in K (required)
...	...	...	...	...

Line	Column	Name	Type	Description
267	A	Air pressure profiles at Season 1, Latitude 1, Level 1	Exp (E10.3)	Air pressure measurements as a function of month, latitude, and vertical level; units in molecules cm^{-3} (required)
	B	Air pressure profiles at Season 1, Latitude 1, Level 2	Exp (E10.3)	Air pressure measurements as a function of month, latitude, and vertical level; units in molecules cm^{-3} (required)
...	...	...	...	...
393	A	Air pressure profiles at Season 1, Latitude 19, Level 1	Exp (E10.3)	Air pressure measurements as a function of season, latitude, and vertical level; units in molecules cm^{-3} (required)
	B	Air pressure profiles at Season 1, Latitude 19, Level 2	Exp (E10.3)	Air pressure measurements as a function of season, latitude, and vertical level; units in molecules cm^{-3} (required)
...	...	...	...	...
1596	A	Air pressure profiles at Season 4, Latitude 19, Level 51	Exp (E10.3)	Air pressure measurements as a function of season, latitude, and vertical level; units in molecules cm^{-3} (required)
1597	A	Average Dobson Values at Latitude 1, Month 1	Real	Average Dobson value as a function of latitude and month (required)
1597	B	Average Dobson Values at Latitude 2, Month 1	Real	Average Dobson value as a function of latitude and month (required)
...	...	...	...	...
1608	A	Average Dobson Values at Latitude 19, Month 12	Real	Average Dobson value as a function of latitude and month (required)

Line	Column	Name	Type	Description
1609	A	Air Temperature at Level 1	Real	Air temperature for a standard atmospheric profile, units in K (required)
	B	Air Pressure at Level 1	Real or Exp	Air pressure for a standard atmospheric profile, units in molecules cm^{-3} (required)
	C	Ozone Concentration at Level 1	Real or Exp	Ozone concentration for a standard atmospheric profile, units in molecules cm^{-3} (required)
	C	Aerosol Attenuation at Level 1	Real or Exp	Aerosol attenuation for a standard atmospheric profile
1659	A	Air Temperature at Level 51	Real	Air temperature for a standard atmospheric profile, units in K (required)
	B	Air Pressure at Level 51	Real or Exp	Air pressure for a standard atmospheric profile, units in molecules cm^{-3} (required)
	C	Ozone Concentration at Level 51	Real or Exp	Ozone concentration for a standard atmospheric profile, units in molecules cm^{-3} (required)
	D	Aerosol Attenuation at Level 51	Real or Exp	Aerosol attenuation for a standard atmospheric profile

6.1.9. TOMS: Total ozone mapping spectrometer data

Used by: JPROC

TOMS is the logical name for the ASCII data file containing total ozone column measurements taken by the Total Ozone Mapping Spectrometer instrument aboard the sun-synchronous polar orbiting Nimbus satellite. The data is presented for specific Julian dates as a function of location on the earth (latitude and longitude).

A detailed description of the file format is provided in Table 4.11 below. The files are fix-formatted.

Table 6.11.

Line	Column	Name	Type	Description
1	A	Julian Day	Int	Julian start day of the file, DDD; preceded by 6 blank spaces (required)
	B	Julian Year	Int	Start year of the file, YYYY; preceded by 9 blank spaces (required)
2		Header	String	80 character line describing the contents of the file (if omitted, needs line placeholder)
3		Header	String	80 character line describing the contents of the file (if omitted, needs line placeholder)
4	A	TOMS Data	Int	TOMS ozone measurements as a function of longitude and latitude, ###; line starts with a space, then space delimited 25 values per line (required)
...	...	...	...	...

6.1.10. O2ABS: Molecular oxygen absorption cross-section data

Used by: JPROC

O2ABS is the logical name for the ASCII data file containing absorption cross-section and quantum yield data for O_2 photolysis. The data in these files are listed as a function of wavelength. These files follow the same format as the CSQY files described above.

6.1.11. O3ABS: Ozone absorption cross-section data

Used by: JPROC

O3ABS is the logical name for the ASCII data file containing absorption cross-section and quantum yield data for O_3 photolysis. The data in this files are listed as a function of wavelength. This files follows the same format as the CSQY files described above.

6.1.12. Meteorological Data Input Files

Used by: MCIP

MCIP reads either MM5 (version 2 or version 3) binary output files (MMOUT) or WRF binary outputs to generate I/O API formatted netCDF files for input to CMAQ and SMOKE. For details about the format of the MMOUT files

visit the MM5 homepage at http://www.mmm.ucar.edu/mm5. For information about the format of the WRF output files visit the WRF homepage at http://wrf-model.org.

6.1.13. BNDY_CONC_1 : Boundary Conditions

Used by: CCTM

CMAQ boundary condition data are of the BNDARY3 file type. Produced by the boundary condition processor, BCON, the CCTM reads these data and correlates them with the interior data by the use of a pointer system. This pointer system designates the beginning location of the data in memory which starts a new side of the domain (e.g., South, East, North, and West). Figure 4.1 illustrates this input data structure and the relationship of the boundary data to the interior data within CMAQ modules.

Each species being modeled should be on the boundary conditions file. If modeled species are not contained in this file, the boundary condition for these species will default to the value 1E-30. The perimeter of the CMAQ domain is NTHIK cells wide where NTHIK = (2*NROW)+(2*NCOL)+4. The CMAQ program BCON (Section 2.2.2) is used to generate CMAQ boundary condition input files from either existing CMAQ outputs or from ASCII-vertical profiles. Figure 4.2 is a graphical example of the CMAQ boundary conditions file.

Figure 6.1. Illustration of CMAQ boundary condition file

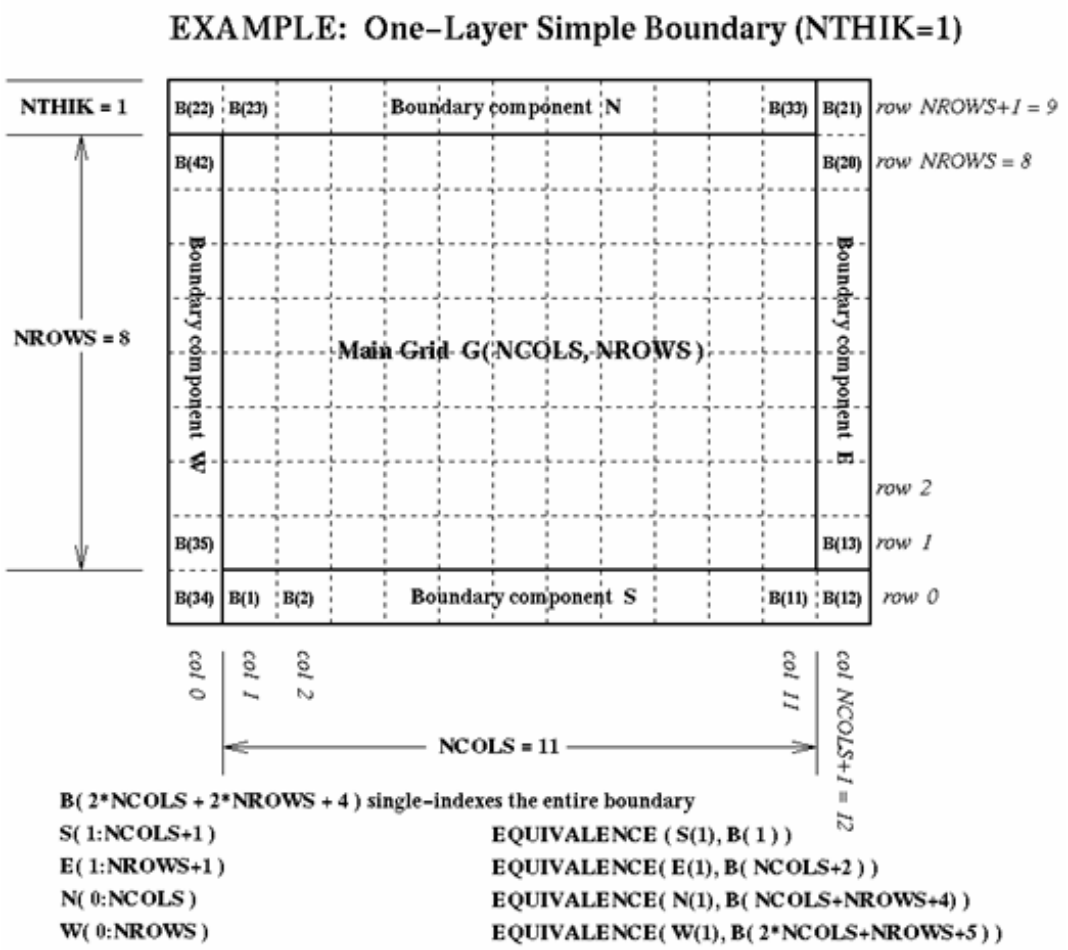
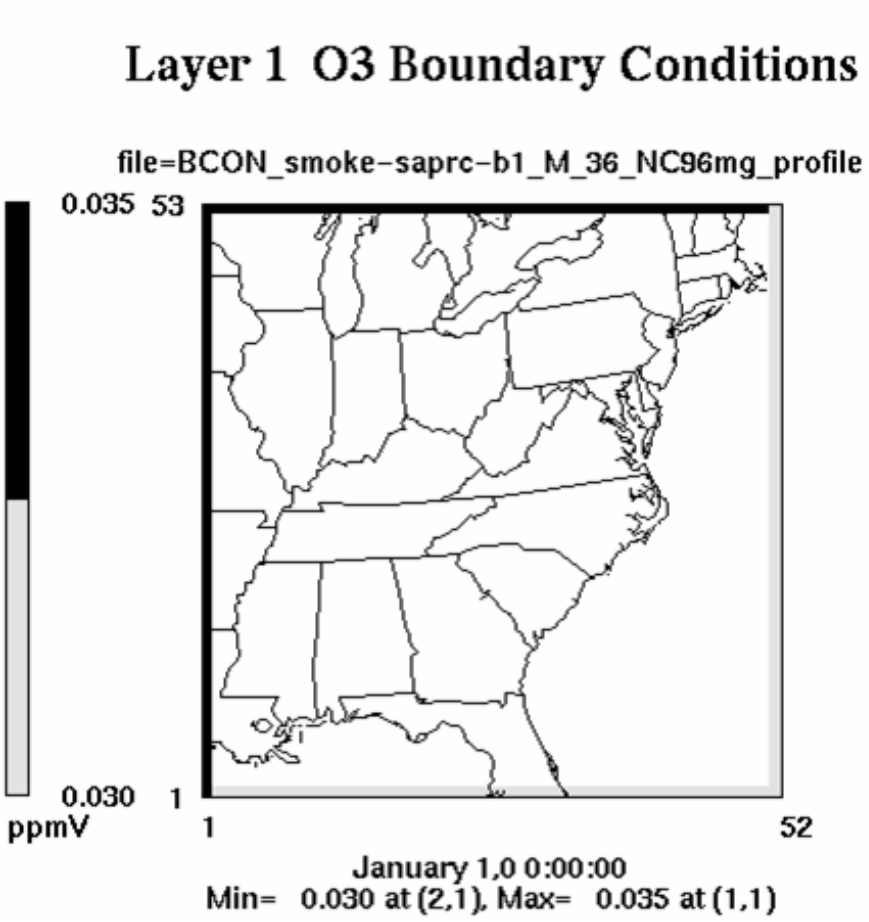


Figure 6.2. Graphical example of a CMAQ gridded boundary conditions file

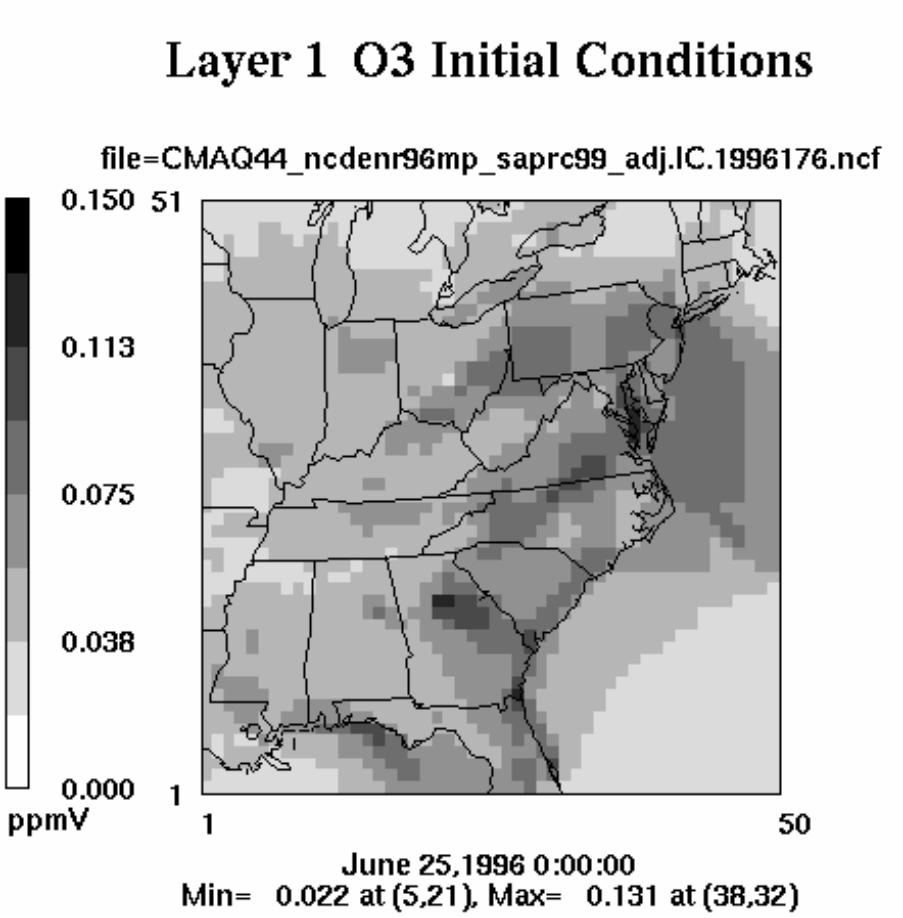


6.1.14. INIT_CONC_1: Initial Conditions

Used by: CCTM

CMAQ requires the initial concentrations of each species being modeled to be input to the model. The initial conditions input file type is GRDDED3 and does not vary with time. The file data is looped accordingly: by column, by row, by layer, by variable. Initial conditions files have the identical structure as concentration files, so as to allow the user to initialize the results of day 2 (as an example) with the predicted concentrations from the last hour of day 1. This allows the CMAQ user flexibility to segment simulations in any way they want. The CMAQ program ICON (Section 2.2.2) is used to generate CMAQ initial condition input files from either existing CMAQ outputs or from ASCII-vertical profiles. Figure 4.3 is a graphical example of the CMAQ initial conditions file.

Figure 6.3. Graphical example of a CMAQ gridded initial conditions file



6.1.15. JTABLE: Photolysis Rates Lookup Table

Used by: CCTM

Each of the three default gas-phase mechanisms in CMAQ (CB-IV, SAPRC99, and RADM2) contain photolysis reactions that require clear sky reaction rates that are precomputed from kinetics data at various altitudes and latitude bands. The CMAQ program JPROC (Section 2.2.3) generates photolysis rate lookup tables for input to CMAQ. Referred to as JTABLE files in CMAQ terminology, these files contain a header section that describes the contents of the file and a data section with clear sky photolysis rates at different times of the day.

The first line of the header contains the Julian date of the data in the file. This date will correspond to the start date for the simulation that inputs the JTABLE data. Four pairs of information follow describing the altitude (m), latitude (degrees), hour angle (from noon), and photolytic reaction name for the data contained in the file. Each data pair begins with a line describing the number of values for each variable and the variable name, followed by the values for each variable. The altitude (variable = LEVELS), latitude (variable = LATITUDES), and hour angle (variable = HOUR ANGLES) variables have fixed values that are hard coded in the program JPROC (routine jproc.F). The reaction names (variable = PHOTOLYTIC REACTIONS) are mechanism-dependent and vary with the choice of gas-phase mechanism.

The data section of the file contains data blocks that are mapped to the header using a three digit code. Each block corresponds to an altitude, latitude, and photolysis reaction and contains nine values of clear sky photolysis rates for the nine hour angles listed in the header. The three digit code maps the altitude/latitude/reaction number to the data block. For example, the data block that uses the code “3 1 2” corresponds to altitude 3, latitude 1, and reaction 2 as

listed in the file header. A detailed description of the JTABLE file format is provided in Table 4.2. The files are list-formatted.

Table 6.12. JTABLE file format description

Line	Column	Name	Type	Description
1	A	JVDATE	String	Julian date of the file; YYYYDDD (required)
	B	Comment	String	Description of the Julian date field (optional)
2	A	JVHT	Int	Number of vertical levels covered by the data (required)
	B	Level Field Indicator	String	LEVELS (required)
	C	Comment	String	Description of the level field, usually units in m (optional)
3	A	XZJV_1	Real	Height of level 1; units in m (required)
	B	XZJV_2	Real	Height of level 2; units in m (required)
...	...	...	...	...
...	x	XZJV_x	Real	Height of level x; units in m (required)
4	A	JVLAT	Int	Number of latitudes covered by the data (required)
	B	Latitude Field Indicator	String	LATITUDES (required)
	C	Comment	String	Description of the latitudes field, usually units in degrees (optional)
5	A	XLATJV_1	Real	Latitude 1; units in degrees (required)
	B	XLATJV_2	Real	Latitude 2; units in degrees (required)
...	...	...	...	...
...	x	XLATJV_x	Real	Latitude x; units in degrees (required)
6	A	JVTMAX	Int	Number of hour angles covered by the data (required)
	B	Hour Angle Field Indicator	String	HOUR ANGLES (required)
	C	Comment	String	Description of the hour angles field, usually units in hours from noon (optional)

Line	Column	Name	Type	Description
7	A	XHAJV_1	Real	Hour angle 1; units in hours from noon (required)
	B	XHAJV_2	Real	Hour angle 2; units in hours from noon (required)
...	...	...	...	...
...	x	XHAJV_x	Real	Hour angle x; units in hours from noon (required)
8	A	JPHOT	Int	Number of photolytic reactions covered by the data (required)
	B	Reaction Field Indicator	String	PHOTOLYTIC REACTIONS (required)
	C	Comment	String	Description of the reactions field (optional)
9	A	PHOT_NM_1	String	Single quote delimited name of photolysis reaction 1 (required)
	B	Delimiter	Char	Comma delimiter separating reaction name from multiplier (required)
	C	ACLD_1	Real	Multiplier for reaction 2 (required)
10	A	PHOT_NM_2	String	Single quote delimited name of photolysis reaction 2 (required)
	B	Delimiter	Char	Comma delimiter separating reaction name from multiplier (required)
	C	ACLD_2	Real	Multiplier for reaction 2 (required)
...	...	...	...	...
x	A	PHOT_NM_x	String	Single quote delimited name of photolysis reaction x (required)
	B	Delimiter	Char	Comma delimiter separating reaction name from multiplier (required)
	C	ACLD_x	Real	Multiplier for reaction x (required)

Line	Column	Name	Type	Description
x + 1	A	NHTO	Int	Vertical level cross-reference to header data (required)
	B	NLATO	Int	Latitude cross-reference to header data (required)
	C	NPHOTO	Int	Photolysis reaction cross-reference to header data (required)
x + 2	A	XJVAL_1	Real or Exp	Clear sky photolysis rate at NHTO, NLATO, NPHOTO, and hour angle 1 (required)
	B	XJVAL_2	Real or Exp	Clear sky photolysis rate at NHTO, NLATO, NPHOTO, and hour angle 2 (required)
	C	XJVAL_3	Real or Exp	Clear sky photolysis rate at NHTO, NLATO, NPHOTO, and hour angle 3 (required)
...	...	...	...	...
	JVTMAX	XJVAL_(JVTMAX)	Real or Exp	Clear sky photolysis rate at NHTO, NLATO, NPHOTO, and hour angle corresponding to JVTMAX (required)
x + 3	A	NHTO	Int	Vertical level cross-reference to header data (required)
	B	NLATO	Int	Latitude cross-reference to header data (required)
	C	NPHOTO	Int	Photolysis reaction cross-reference to header data (required)
x + 4	A	XJVAL_1	Real or Exp	Clear sky photolysis rate at NHTO, NLATO, NPHOTO, and hour angle 1 (required)

Line	Column	Name	Type	Description
	B	XJVAL_2	Real or Exp	Clear sky photolysis rate at NHTO, NLATO, NPHOTO, and hour angle 2 (required)
	C	XJVAL_3	Real or Exp	Clear sky photolysis rate at NHTO, NLATO, NPHOTO, and hour angle 3 (required)
...	...	...	...	...
	JVTMAX	XJVAL_(JVTMAX)	Real or Exp	Clear sky photolysis rate at NHTO, NLATO, NPHOTO, and hour angle corresponding to JVTMAX (required)
...	...	...	...	...

A sample of the important sections of a JVAL file is shown below.

```
 1999181 (yyyyddd) Julian Date for the file
 7 LEVELS (m)
   0.0 1000.0 2000.0 3000.0 4000.0 5000.0 10000.0
 6 LATITUDES (deg)
   10.0 20.0 30.0 40.0 50.0 60.0
 9 HOUR ANGLES (from noon)
   0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0
 6 PHOTOLYTIC REACTIONS
   'NO2_CBIV88     ', 1.0
   'O3O1D_CBIV88   ', 1.0
   'HCHOmol_CBIV88 ', 1.0
   'HCHOrad_CBIV88 ', 1.0
   'ALD_CBIV88     ', 1.0
   'ACROLEIN       ', 1.0
 1 1 1
5.0964761E-01 4.9923715E-01 4.6422747E-01 4.0129572E-01 3.0394882E-01
1.6590215E-01 3.2829735E-02 0.0000000E+00 0.0000000E+00
```

6.1.16. EMISfile: Emissions

Used by: CCTM

CMAQ can accept emissions inputs from a variety of emissions models and preprocessors. The preferred option is the Sparse Matrix Operator Kernel Emissions (SMOKE) modeling system which is a collection of programs that efficiently process emissions data for input to air quality models.

The emissions file sorts the emitted gas-phase and aerosol species by grid cell and time. The file type is GRDDED3 and the units are in moles/s. The file data is looped as follows: by column, by row, by layer, by variable, and by input time step. CMAQ does not artificially distinguish between surface and elevated emissions sources; elevated sources are provided to CMAQ as vertically resolved emissions. At present all emissions estimates are contained within a single input emission file for each day. Future versions of CMAQ may allow the online calculation of the vertical plume rise and the addition of process-based emissions, such as sea salt or windblown dust.

6.1.17. GC2file: Two-dimensional Grid Cross Product Input (GRID_CRO_2D)

Used by: CCTM

CMAQ requires the following variables to be present in the Grid Cross 2D meteorology file.

LAT:	latitude in degrees (northern hemisphere is positive)
LON:	longitude in degrees (western hemisphere is negative)
MSFX2:	squared map-scale factor (Cross; cell interfaces)
HT:	terrain elevation (m)
DLUSE:	dominant land use (category)

6.1.18. GD2file: Two-dimensional Grid Dot Product Input (GRID_DOT_2D)

Used by: CCTM

CMAQ requires the following variables to be present in the Grid Dot 2D meteorology file.

LAT:	latitude in degrees (northern hemisphere is positive)
LON:	longitude in degrees (western hemisphere is negative)
MSFD2:	squared map scale factor (Dot; cell centers)

6.1.19. MB3file: Three-dimensional Meteorological Boundary Input (MET_BDY_3D)

Used by: CCTM

CMAQ requires the following variables to be present in the Met Boundary 2D meteorology file.

JACOBF:	total Jacobian at layer face (m)
JACOBM:	total Jacobian at layer middle (m)
DENSA_J:	J weighted total air density (kg/m^3)
WHAT_JD:	J and density weighted vertical contra-W (kg/m-s)

TA:	air temperature (K)
QV:	mixing ratio (kg water/kg air)
PRES:	air pressure (pascals)
DENS:	air density (kg/m^3)
WWIND:	vertical wind velocity (m/s)
ZH:	mid-layer height above ground [hydrostatic only] (m)
ZF:	full layer height above ground [hydrostatic only] (m)
QC:	cloud water mixing ratio (kg/kg)
QR:	rain water mixing ratio (kg/kg)

6.1.20. MC2file: Two-dimensional Meteorological Cross-Product Input (MET_CRO_2D)

Used by: CCTM

CMAQ requires the following variables to be present in the Met Cross 2D meteorology file.

PRSFC:	surface pressure (pascals)
JACOBS:	total Jacobian at surface (m)
USTAR:	cell averaged horizontal friction velocity (m/s)
WSTAR:	convective velocity scale (m/s)
PBL:	planetary boundary layer height (m)
ZRUF:	surface roughness length (m)
MOLI:	inverse Monin-Obukhov length (1/m)
QFX:	latent heat flux (Watts/m^2)
HFX:	sensible heat flux (Watts/m^2)
RADYNI:	inverse aerodynamic resistance (m/s)
RBNDYI:	inverse laminar boundary layer resistance (m/s)
RSTOMI:	inverse bulk stomatal resistance (m/s)
TEMPG:	skin temperature at ground (K)
TEMP10:	10 meter temperature (K)
TEMP1P5:	1.5 meter temperature (K)
WSPD10:	10 meter windspeed (m/s)
WDIR10:	10 meter wind direction (m/s)
GLW:	longwave radiation at ground (Watts/m^2)
GSW:	solar radiation absorbed at ground (Watts/m^2)
RGRDN:	solar radiation reaching the surface (Watts/m^2)
RN:	accumulated non-convective precipitation (cm)
RC:	accumulated convective precipitation (cm)
CFRAC:	total cloud fraction (fraction)

WBAR_KF:	average liquid water content of clouds (g/m^3)
CLDT_KF:	cloud top (m)
CLDB_KF:	cloud bottom (m)
SNOCOV:	snow cover (1 = yes, 0 = no)
VD_SO2:	deposition velocities for deposited species SO_2 (m/s)
VD_SULF:	deposition velocities for deposited species SULF (m/s)
VD_NO2:	deposition velocities for deposited species NO_2 (m/s)
VD_NO:	deposition velocities for deposited species NO (m/s)
VD_O3:	deposition velocities for deposited species O_3 (m/s)
VD_HNO3:	deposition velocities for deposited species HNO_3 (m/s)
VD_H2O2:	deposition velocities for deposited species H_2O_2 (m/s)
VD_ALD:	deposition velocities for deposited species ALD (m/s)
VD_HCHO:	deposition velocities for deposited species HCHO (m/s)
VD_OP:	deposition velocities for deposited species OP (m/s)
VD_PAA:	deposition velocities for deposited species PAA (m/s)
VD_ORA:	deposition velocities for deposited species ORA (m/s)
VD_NH3:	deposition velocities for deposited species NH_3 (m/s)
VD_PAN:	deposition velocities for deposited species PAN (m/s)
VD_HONO:	deposition velocities for deposited species HONO (m/s)
VD_CO:	deposition velocities for deposited species CO (m/s)
VD_METHANOL:	deposition velocities for deposited species METHANOL (m/s)
VD_N2O5:	deposition velocities for deposited species N_2O_5 (m/s)
VD_NO3:	deposition velocities for deposited species NO_3 (m/s)
VD_GEN_ALD:	deposition velocities for deposited species GEN_ALD (m/s)

6.1.21. MC3file: Three-dimensional Meteorological Cross-Product Input (MET_CRO_3D)

Used by: CCTM

CMAQ requires the following variables to be present in the Met Cross 3D meteorology file.

JACOBF:	total Jacobian at layer face (m)
JACOBM:	total Jacobian at layer middle (m)
DENSA_J:	J weighted total air density (kg/m^3)
WHAT_JD:	J and density weighted vertical contra-W (kg/m-s)
TA:	air temperature (K)
QV:	mixing ratio (kg water/kg air)

PRES:	air pressure (pascals)
DENS:	air density (kg/m^3)
WWIND:	vertical wind velocity (m/s)
ZH:	mid-layer height above ground [hydrostatic only] (m)
ZF:	full layer height above ground [hydrostatic only] (m)
QC:	cloud water mixing ratio (kg/kg)
QR:	rain water mixing ratio (kg/kg)

6.1.22. MD3file: Three-dimensional Meteorological Dot-Product Input (MET_DOT_3D)

Used by: CCTM

CMAQ requires the following variables to be present in the Met Dot 3D meteorology file.

UWIND:	u component of horizontal wind (m/s)
VWIND:	v component of horizontal wind (m/s)
UHAT_JD:	contra-U*Jacobian*Density (kg/m-s)
VHAT_JD:	contra-V*Jacobian*Density(kg/m-s)

6.2. Basic CMAQ Output Files

As all of the CMAQ programs produce output files that are input to other CMAQ programs, differentiating between CMAQ input and output is not entirely clear. The previous section dealt with the output files from JPROC, ICON, BCON, and MCIP as inputs to the CCTM. In this section, we will detail the general structure of all CMAQ binary outputs and provide details on the CCTM output files.

All CMAQ programs, other than JPROC, produce output files that adhere to the IO API format (Chapter 8); JPROC produces ASCII output files. The IO API formatted CMAQ output files are 3-dimensional gridded, time-stepped binary files that contain headers with metadata describing the file contents. These machine-independent and network transparent binary files are transferable between different computer architectures. The advantages of using the IO API output format include (Coats, 2005),

- performance: visualization and analysis programs looking at selected parts of the data don't need to read unrequested data from the files.
- modularity: data can be read or written in any order (or not at all). The same input files can serve both simple engineering models and full-chemistry models -- the former reading just a few of the variables from the files, the latter reading most of them. The modular-model structure used for CMAQ depends upon this feature.
- robustness: data are "tagged" by name, date, and time; miscounted record-numbers don't scramble temperatures with pressures, for example.

The IO API output data are written to files using calls to the WRITE3 logical IO/API function. The amount of information saved to CMAQ output files is defined almost entirely by the developer of a particular subroutine. Should a user

decide additional (or less) output is needed, simple code changes could be completed to add (or remove) the selected data. For example, in the current release version of CMAQ (version 4.5) the coefficients of vertical eddy diffusivity are not written to any output file. However, the physical Kv values are calculated in the model in the subroutine EDYINTB and contravariant diffusivities are computed in the VCONTDIFF subroutine. It would be a relatively simple matter to insert code to output the values, either as a new individual output file or as an appendage to an existing file.

In addition to model data output, CMAQ can optionally produce log files containing the standard output from the different CMAQ processors. If the log file option is not selected by the user, CMAQ will write all of the log information to the screen along with the standard error, which can be captured to a text file using basic Unix syntax.

6.2.1. CMAQ Output Log

All of the CMAQ processors generate standard output and standard error during execution. For all of the processors other than the CCTM, this diagnostic output information can be captured to a log file at execution using a Unix redirect command. For example to capture the standard output and error of a BCON simulation, use the following command:

```
run.bcon.csh >& run.bcon.log
```

For the CCTM the "LOGFILE" environment variable allows users to specify the name of a log file for capturing the standard output from the program. If this variable is not set, the standard output is written to the terminal and can be captured using the Unix redirect command (">"), as shown in the example above.

6.2.2. CONC: CCTM hourly instantaneous concentration file

The 3-D CMAQ hourly concentration file (CONC) is the most commonly referenced CMAQ output file. Containing gas-phase species mixing ratios (ppmV) and aerosol species concentrations (µg/m3), CONC files include instantaneous model species concentrations at the end of each model hour. The number and types of species contained in the CONC files is dependent on the chemical mechanism and aerosol model configurations selected when compiling the CCTM. The species concentration include files (CONC.EXT) within the mechanism include directories list the species that are written to the CONC files for each mechanism configuration. The GC_CONC.EXT file lists the gas-phase species and the AE_CONC.EXT file lists the aerosol species written to the CONC file. You can remove species from these CONC.EXT files to reduce the number of species written to the CONC file.

6.2.3. ACONC: CCTM hourly average concentration file

The 2-D CMAQ integral average concentration file (ACONC) includes average model species concentrations for each model hour, as opposed to instantaneous concentrations at the end of each output time step. The species written to the ACONC file are set by the user in the CCTM run script using the variable AVG_CONC_SPCS. The model layers across which to derive the integral average concentration are also set in the CCTM run script using the variable ACONC_BLEV_ELEV.

6.2.4. DRYDEP: CCTM hourly cumulative dry deposition file

The 2-D CMAQ dry deposition file (DRYDEP) includes instantaneous hourly dry deposition fluxes (kg/hectare) for selected model species. CMAQ calculates dry deposition for all of the species listed in the dry deposition include files within the mechanism include directories. Dry deposition include files exist for gas-phase species (GC_DDEP.EXT), aerosol species (AE_DDEP.EXT), and inert model species (NR_DDEP.EXT). You can remove species from these

DDEP.EXT files to adjust the number of species that undergo the dry deposition process and are written to the DRYDEP output file.

6.2.5. WETDEP: CCTM hourly cumulative wet deposition file

The 2-D CMAQ wet deposition file (WETDEP) includes cumulative hourly wet deposition fluxes (kg/hectare) for selected model species. CMAQ calculates wet deposition for all of the species listed in the wet deposition include files within the mechanism include directories. Wet deposition include files exist for gas-phase species (GC_WDEP.EXT), aerosol species (AE_WDEP.EXT), and inert model species (NR_WDEP.EXT). You can remove species from these WDEP.EXT files to adjust the number of species that undergo the wet deposition process and are written to the WETDEP output file.

6.2.6. AEROVIS: CCTM hourly instantaneous visibility metrics

The 2-D CMAQ visibility file (AEROVIS) contains hourly Mie and reconstructed visual range (km^{-1}) and extinction coefficients (deciviews).

6.3. Diagnostic and advanced CMAQ Output Files

Along with the basic outputs detailed in the previous section, CMAQ can be configured to output several auxiliary files for diagnosing model performance and for plume-in-grid treatment of large point sources.

6.3.1. AERODIAM: Instantaneous hourly aerosol diameter file

6.3.2. WETDEP2: CCTM cloud diagnostics file

6.3.3. PING: Plume-in-Grid unmerged plume data file

6.3.4. PINGDRYDEP: Plume-in-Grid dry deposition file

6.3.5. PINGAERODIAM: Plume-in-Grid hourly aerosol diameter file

6.3.6. PA: Process analysis output - integrated process rate file

6.3.7. IRR: Process analysis output - integrated reaction rates

Chapter 7. Developing Meteorological Fields

Table of Contents

7.1. Credits and Disclaimers For Use of MM5 135
7.2. System Requirements 136
7.3. Software Languages 136
7.4. Data Requirements 136
7.5. Meteorology Model Pre-Processing 138
7.6. Meteorology Model (MM5) 139
7.7. Meteorology Model Post-Processing 142
7.8. Meteorology Model Visualization and Evaluation 142
7.9. Model Training and User Support 143
7.10. Using MM5 with CMAQ 143
7.11. References 143

Meteorological data are important in many of the processes simulated in the CMAQ model. The meteorology model that has been selected and evaluated as part of Models-3 to supply this data is the MM5. All of the software in the Penn State/National Center for Atmospheric Research (NCAR) mesoscale modeling system has been dedicated to the public domain and is presently used for both research and operational purposes by many organizations. MM5 was selected to support Models-3 and CMAQ because it is a state-of-the-science meteorological model that is in the public domain of the scientific community. Improvements to MM5 within the meteorological community are on-going, and these enhancements can be evaluated for their applicability to air quality modeling.

7.1. Credits and Disclaimers For Use of MM5

MM5 was developed in cooperation with Pennsylvania State University and the University Corporation for Atmospheric Research (UCAR). Penn State and UCAR make no proprietary claims, either statutory or otherwise, to this version and release of MM5, and consider MM5 to be in the public domain for use by any person or entity without any fee or charge. Penn State and UCAR shall be credited in any publications that result from the use of MM5. The names “Penn State” and “UCAR” shall not be used or referenced in any advertising or publicity that endorses or promotes any products or commercial entity associated with or using MM5, or any derivative works thereof, without the written authorization of UCAR and Penn State.

MM5 is provided to the public by Penn State and UCAR on an “as is” basis, and any warranties, either express or implied, including but not limited to implied warranties of non-infringement, originality, merchantability, and fitness for a particular purpose, are disclaimed. Neither UCAR nor Penn State will be obligated to provide support, consulting, training, or assistance of any kind with regard to the use, operation, and performance of MM5, nor to provide any updates, revisions, new versions, error corrections or “bug” fixes. In no event will UCAR and Penn State be liable for any damages, whatsoever, whether direct, indirect, consequential, or special, which may result from an action in contract, negligence, or other claim that arises out of or in connection with the access, use, or performance of MM5, including infringement actions.

Data that are obtained from NCAR (e.g., global analyses and observations) are prepared and maintained by the Data Support Section, Scientific Computing Division, NCAR. NCAR is operated by UCAR and is sponsored by the National Science Foundation. Published research that uses data from the NCAR archive shall credit the maintainers of the archive, and the original source(s) of the data shall be acknowledged.

7.2. System Requirements

MM5, which superseded its predecessor MM4, was released to the public in the early 1990s. MM5 and its supporting software (i.e, pre-processing and post-processing programs) were originally designed to run on a Cray supercomputer. Over the past few years, the software structure has been modified to enable the model to also run on various workstations. Today, MM5 can run on many hardware platforms including Cray, SGI, Sun, DEC, Hewlett Packard, IBM, and personal computers running Linux. In addition, MM5 has software directives to efficiently use multi-processor computers, including shared-memory, distributed-memory, and massively parallel architectures. Although MM5 has been run on workstations, not all of the supporting software is available in a workstation version at this time. Also, there may be significant performance differences when running MM5 on different manufacturers' workstations and supercomputers.

CMAQ Version 4.4 does **not** include MM5. Users will need to download and install MM5 on their own.

7.3. Software Languages

MM5 and its supporting software are largely written in Fortran 77. Some of the software has been converted to include features in Fortran 90. A C pre-processor is used to compile the software with directives for various hardware platforms. All of the MM5 system software is compiled using the "make" utility. The scripts that support the execution of various programs in the software system are written in C-shell.

7.4. Data Requirements

Several datasets are required to run MM5. Those datasets can be classified as "static" (data that are representative of the simulation domain, regardless of the episode period) and "case-specific" (data that are specific to the episode).

7.4.1. Static Data

Certain static data bases need to be available to define the simulation domain for MM5. There are currently five resolutions of terrain data bases and three resolutions of land-use data bases that NCAR distributes to support MM5 simulations. The terrain data bases are 60-minute (1 degree or about 111 km in mid-latitudes), 30-minute (0.5-degree or about 55 km in mid-latitudes), 10-minute (about 18.5 km in mid-latitudes), 5-minute (about 9.25 km in mid-latitudes), and 1-minute (about 0.925 km in mid-latitudes). The first four terrain data bases include world-wide coverage; the 1-minute data base is currently only available for the continental U.S. The land-use data bases are available in 60-minute, 30-minute, and 10-minute resolutions for world-wide coverage. The land-use data includes 13 categories of land use, and each land-use category has a seasonally parameterized set of physical properties that are used to model physics calculations (e.g., albedo, available moisture, emissivity, roughness length, thermal inertia). The terrain and land-use data bases are available in both Cray and IEEE formats from the NCAR anonymous FTP site (ftp://ftp.ucar.edu, in directory /mesouser/Data). The terrain and land-use data are interpolated to the MM5 simulation grid in the TERRAIN program (discussed in Section 3.5.1).

The data bases are assigned to MM5 simulations based on the resolution of the simulation domain. The terrain and land-use data bases for an individual domain should be the same resolution wherever possible (e.g., do not assign a 60-minute terrain and a 30-minute land use to the same domain). Note that higher-resolution terrain and land-use data should be used for higher-resolution domains, even within the same simulation. For example, a 108-km domain may be assigned the 60-minute terrain and land use, while a 36-km nest may be assigned the 30-minute terrain and land use.

7.4.2. Case-Specific Data

Three primary types of case-specific data are required for an MM5 simulation: meteorological background fields, meteorological observations, and surface meteorological fields.

7.4.2.1. Meteorological Background Fields

The meteorological background fields that are used for an MM5 simulation are extracted from a larger-scale (either global or regional) simulation that geographically covers the MM5 simulation domain and simulation period entirely. The background fields are used to generate MM5's "first guess" and lateral boundary conditions. The first guess is improved with synoptic and mesoscale observations to generate an analysis that is transformed into an initial state for the model. The lateral boundary conditions provide forcing to the model throughout the simulation, and they can be especially influential in the later stages of the simulation.

Since CMAQ will be used to simulate very high-resolution domains, it is important to select a high-quality background dataset for the model. The MM5 system can ingest and process global analyses data generated by either the European Centre for Medium-Range Weather Forecasts (ECMWF) or the National Centers for Environmental Prediction (NCEP). With an account on the NCAR Cray or other arrangements with NCAR (see http://www.mmm.ucar.edu/mm5/faqMM5.html), you may retrieve the pre-formatted global analyses data for use in MM5. ECMWF global analyses data is twice daily at a 2.5-degree resolution from January 1980 to December 1989. A special dataset of the European Centre for Medium-Range Tropical Ocean and Global Atmospheric program (ECMTOGA) global analyses is available for restricted usage in MM5. This dataset (at a 2.5-degree resolution from January 1985 through December 1997) is available from NCAR for a fee. NCEP global tropospheric analyses data (twice daily at a 2.5-degree resolution from July 1976 to April 1997) is also available from NCAR. Background analyses data is also available from sources other than NCAR. If you acquire such data from a source other than NCAR, you may need to write software to format the data into a form that REGRID expects, or you may need to modify REGRID to accept the analyses data.

7.4.2.2. Meteorological Observations

Standard meteorological observations of temperature, relative humidity, height, sea-level pressure, and wind are required to improve the background fields with synoptic and mesoscale details. The analysis model (RAWINS, discussed in Section 4.5.3) is set up to process surface observations (including ship reports and buoys) and upper-air observations (including radiosonde observations and wind profiles). Other types of data may be included in the analysis if the appropriate modifications are made to the software.

With an account on the NCAR Cray or other arrangements with NCAR (see http://www.mmm.ucar.edu/mm5/faqMM5.html), you may retrieve pre-formatted surface and upper-air observations that were acquired by NCAR from the National Centers for Environmental Prediction (NCEP). Daily surface observations (including both land and water observations) are available for the period from January 1978 through today. Some of the dates include two sets of files: six-hour, and three-hour. The six-hour files include data observed at 00, 06, 12, and 18 Universal Time Coordinates (UTC). The three-hour files include data observed at 03, 09, 15, and 21 UTC. The more current surface observation files have all observed surface data merged into a single file. The upper-air data are available for the period from January 1985 to April 1997 for 00 UTC and 12 UTC.

If you acquire your meteorological observation data from a source other than NCAR (e.g., from special field studies), you may need to write software to format the observations into a form that RAWINS expects.

7.4.2.3. Meteorological Surface Fields

MM5 requires analyses data for sea-surface temperature and snow to more accurately simulate the meteorological conditions. With an account on the NCAR Cray or other arrangements with NCAR (see http://www.mmm.ucar.edu/mm5/faqMM5.html), you may retrieve such data. If you acquire sea-surface temperature and snow analyses data from a source other than NCAR, you may need to write software to convert the data into a form that REGRID (discussed in Section 3.5.2) expects or modify REGRID to accept the data.

NCAR has sea-surface temperature data from the U.S. Navy and NCEP, as well as sea-surface temperature data generated from climatological data. The Navy sea-surface temperature data is 63 x 63 Northern Hemisphere polar data. It is available for cases between November 1961 and December 1993. More current Navy sea-surface temperatures

may be available from NCAR but are not listed in the NCAR catalog. The NCEP sea-surface temperature data is included as part of the global analysis data and is available at 2.5-degree resolution. The climatological sea-surface temperature data available from NCAR is at 2-degree resolution. When possible, it is more advantageous to use the measured and analyzed sea-surface temperatures rather than the climatology data. The measured and analyzed sea-surface temperatures are generally more representative of the actual field during the simulation. The use of measured sea-surface temperatures can be particularly important (e.g., in an El Niño year) when the actual temperatures deviate strongly from “normal” (e.g., climatology).

The snow data base at NCAR includes Northern Hemisphere analyses from NCEP. The snow analysis is updated about once a week, or following a significant snow event. If you are simulating a summer episode, you should be able to bypass the need for a snow analysis, and MM5 will assume that there is no snow cover in the simulation domain.

7.5. Meteorology Model Pre-Processing

Before the meteorology model can be run, several pre-processing programs must be run to set up the domain for the simulation and to generate a set of initial and boundary conditions for the meteorology model. The pre-processing programs are briefly described in this section.

7.5.1. Defining the Simulation Domain (TERRAIN)

Domains for the meteorology simulations are defined by several primary parameters: number of grid points in each horizontal dimension, grid spacing, center latitude, center longitude, map projection (Mercator, Lambert conformal, or polar stereographic), and number of “nested” domains and their horizontal dimensions. (Some other user-specific parameters are also defined based on the primary parameters, as required.) These parameters are processed by software that is executed only when a new domain location is required. This software, TERRAIN, makes use of high-resolution global terrain and land-use datasets to create “static files” for the domain. The static files currently include values at each grid point for terrain height and land-use specification (e.g., deciduous forest, desert, water). Future releases of TERRAIN with Models-3 may include other time-independent (diurnal and seasonal) and location-specific information. TERRAIN is described by Guo and Chen [1994].

With CMAQ, you can select a (rectangular) domain of any horizontal dimensions, any grid spacing, and most geographical locations. A nesting capability is also available. The number of grid points in the nests is restricted to a multiple of 3, plus 1, if you plan to use a two-way nesting option (discussed in Section 4.6.5). For example, a valid number of grid points for one horizontal dimension can be 100, which is (33 x 3 + 1), or 85, which is (28 x 3 + 1).

7.5.2. Processing the Meteorological Background Fields (REGRID)

After the simulation domain has been established, REGRID is run to generate first-guess fields for the model simulation by horizontally interpolating a larger-scale dataset (global or regional coverage) to the simulation domain. REGRID interpolates the background fields to the simulation domain for times throughout the simulation period; these files are used to ultimately generate lateral boundary conditions for the coarse-domain simulations (nested domains use lateral boundaries from the coarse domain). REGRID also processes the sea-surface temperature and snow files by interpolating the analyses data to the simulation domain. As a last step, REGRID calculates map-scale factors and Coriolis parameters at each grid point to be used by MM5. REGRID is described by Manning and Haagenson [1992].

7.5.3. Objective Analysis (RAWINS)

The program RAWINS performs an objective analysis by blending the first guess fields generated by REGRID with upper-air and surface observations. There are four objective analysis techniques available in RAWINS: circular-shaped (Cressman) weighting function, elliptical-shaped weighting function, banana-shaped weighting function, and multi-

quadric interpolation. In addition, RAWINS performs some data quality checks against user-defined thresholds. The objective analysis is performed for a user-defined number of pressure surfaces (generally mandatory levels plus some additional surfaces). RAWINS is also used to prepare the analyses for the analysis nudging in the forecast model (see Section 4.6.6). RAWINS is described in Manning and Haagenson [1992].

7.5.4. Setting the Initial and Boundary Conditions (INTERPF)

The INTERPF program sets the initial and boundary conditions for the meteorology simulation. The analysis from RAWINS is interpolated from its native vertical coordinate (pressure) to the forecast model's vertical coordinate (a terrain-following “sigma” coordinate). In addition, the state variables are converted as necessary, e.g., relative humidity to specific humidity. Also, INTERPF interpolates the analysis fields for the MM5 staggered grid configuration (discussed in Section 3.6.2) and interpolates the lateral boundary conditions to the sigma coordinate and staggered grid.

7.6. Meteorology Model (MM5)

The following subsections provide a brief overview of the science and options that are available in MM5. A description of the standard MM5 options is found in Grell et al. [1995]. The source code for MM5 is documented in Haagenson et al. [1994].

7.6.1. Brief History

MM5 is the fifth-generation Penn State/NCAR mesoscale model. It has grown from the model used by Anthes in the early 1970s that was documented in Anthes and Warner [1978]. The improvements in MM5 over the previous version (MM4) include the option for non-hydrostatic physics, as well as more sophisticated explicit moisture, boundary layer processes, radiation, convective parameterization, among other improvements. Dudhia [1993] documents the major changes from MM4 to MM5. Starting with MM5 Version 2 (MM5v2), the software has been restructured to run on various hardware platforms in addition to the Cray, with emphasis placed on workstation-based MM5 simulations. The latest is MM5 Version 3 (MM5v3) and is the recommended version for CMAQ simulations.

7.6.2. Horizontal and Vertical Grid

The coordinate system for MM5 is (x, y, sigma). The x and y are a regular lattice of equally spaced points (delta-x = delta-y = horizontal grid spacing in kilometers) forming rows and columns. Sigma is a terrain-following vertical coordinate that is a function of the pressure at the point on the grid (in hydrostatic runs) or the reference state pressure (in non-hydrostatic runs), the surface pressure at the grid point, and the pressure at the top of the model. Sigma varies from 1 at the surface to 0 at the top of the model. The influence of the terrain on the sigma structure diminishes with height, so that the sigma surfaces near the top of the model are nearly parallel.

See Figure 5.1 for a sample sigma structure of 16 vertical layers (17 full-sigma levels). Dashed lines denote half-sigma levels, and solid lines denote full-sigma levels. Vertical velocity (ω) is calculated on full-sigma levels. All other prognostic variables (a) are calculated on half-sigma levels. The figure (based on Dudhia et al., [1998]) is not drawn to scale. The vertical grid in MM5 is staggered, and all of the prognostic meteorological variables, except for vertical motion, are calculated on half-sigma surfaces. Each half-sigma is located in space approximately half the distance between two full-sigma levels.

Figure 7.1. Example sigma vertical structure in MM5.

The horizontal grid in MM5 has an Arakawa-B staggering of the velocity vectors with respect to the scalars [Arakawa and Lamb, 1977]. The momentum variables (u- and v-components of wind and the Coriolis force) are on “dot” points, while all other variables (e.g., mass and moisture variables) are on “cross” points. The dot points form the regular lattice for the simulation domain, while the cross points are offset by 0.5 grid point in both the x and y directions. Refer to Figure 5.2 for a sample Arakawa-B staggering. Note that the interpolation of the variables to the staggered grid is done automatically within the INTERP program. The smaller inner box is a representative mesh staggering for a 3:1 coarse-grid distance to fine-grid distance ratio.

Figure 7.2. Sample Arakawa-B staggering.

7.6.3. Prognostic Equations

MM5 is based on primitive physical equations of momentum, thermodynamics, and conservation. The state variables are temperature, specific humidity, grid-relative wind components, and pressure. In the prognostic equations, the state variables are mass-weighted with a modified surface pressure. MM5 can be run as either a hydrostatic or non-hydrostatic model. In the hydrostatic model, the state variables are explicitly forecast. In the non-hydrostatic model [Dudhia, 1993], pressure, temperature, and density are defined in terms of a reference state and perturbations from the reference state. The vertical (sigma) coordinate is defined as a function of pressure.

7.6.3.1. Time Differencing

The hydrostatic and non-hydrostatic versions of MM5 use different time differencing schemes to filter the fast waves from the prognostic solutions in the model. In the non-hydrostatic model, a semi-implicit scheme based on Klemp and Wilhelmson [1978] is used to filter the acoustic waves from the model solution. In the hydrostatic model, a split-explicit scheme based on Madala [1981] is used to filter gravity waves from the model solution. The time differencing in MM5 is extensively discussed in Grell et al. [1995.

7.6.3.2. Lateral Boundary Conditions

There are five options for lateral boundary conditions in MM5: fixed, relaxation, time dependent, time dependent and inflow/outflow dependent, and sponge. The lateral boundaries in MM5 consist of the outer five grid points on the horizontal perimeter of the simulation domain. (The outer four grid points are used for boundary conditions for “cross” point variables.) The lateral boundary conditions for the coarse domain are derived from the meteorological background fields processed in REGRID and INTERP. For the nests (when the one-way nest option is selected), the lateral boundary conditions are interpolated from the simulation on the parent domain.

7.6.4. Model Physics

Several model physics options are available to users of MM5. They are briefly noted below. The model physics options are further discussed and compared in Dudhia et al. [1998].

7.6.4.1. Radiation

There are four atmospheric radiation cooling schemes available in MM5: none, simple cooling, the Dudhia long and short wave radiation scheme, and the NCAR/CCM2 radiation scheme.

7.6.4.2. Convective Parameterization

There are currently six convective parameterization schemes available in MM5: Anthes-Kuo, Fritsch-Chappel, Arakawa-Schubert, Kain-Fritsch, Betts-Miller, and Grell. These convective parameterization schemes have been designed for

use at various simulation scales, and they are not interchangeable. For example, each scheme uses different assumptions for convective coverage on the sub-grid-scale and for the convective trigger function. The convective parameterization schemes also differ greatly in CPU usage and memory requirements.

The option for no convective parameterization for the domain (e.g., explicitly resolved convection on the grid scale) is available in MM5. This option is generally used for simulations on domains with horizontal grid spacing smaller than 10 km.

7.6.4.3. PBL Processes

Six PBL parameterization schemes are available in MM5: bulk formula, Blackadar, Burk-Thompson, MRF (Hong-Pan), Eta, Gayno-Seaman and Pleim-Xiu. These parameterizations differ the most in the turbulent closure assumptions that are used. The PBL parameterization schemes also differ greatly in CPU usage and median range forecast requirements.

7.6.4.4. Surface Layer Processes

The surface layer processes in MM5 have been parameterized with fluxes of momentum, sensible heat, and latent heat, following Zhang and Anthes [1982]. The energy balance equation is used to predict the changes in ground temperature. The 13 land-use categories are used to seasonally define the physical properties at each grid point (e.g., albedo, available moisture, emissivity, roughness length, and thermal inertia). In addition, a five-layer soil temperature model is available but can only be used in conjunction with the Blackadar and MRF PBL schemes.

7.6.4.5. Resolvable-Scale Microphysics Schemes

There are six resolvable-scale (explicit grid-scale) microphysics schemes available in MM5: removal of supersaturation, the Hsie warm rain scheme, the Dudhia simple ice scheme, the Reisner mixed phase scheme, the Reisner mixed phase scheme with graupel, and the NASA/Goddard microphysics with hail/graupel. These microphysics schemes have been designed with varying degrees of complexity for different applications of the model. In addition, there are new prognostic output variables that are generated by the more sophisticated schemes. These microphysics schemes also differ greatly in CPU usage and memory requirements.

7.6.5. Nesting

MM5 has the capability to simulate nested domains of finer resolutions within the primary simulation domain. In MM5, the software is configured to enable up to 9 nests (10 domains) within a particular run. However, due to current hardware resources, the state of the science, numerical stability, and practicality, the number of domains in a simulation is generally limited to four or fewer.

Nesting can be accomplished by either “one-way” or “two-way” methods. In one-way nesting, the coarse-resolution domain simulation is run independently of the nest. That domain then provides the initial and boundary conditions for its nest. There are no feedbacks between the coarse domain and its nest in one-way nesting. Note that the simulated meteorology at the same grid point in the coarse-resolution domain is likely to be different (if only slightly) from the nest in a one-way nest simulation.

Two-way interactive nesting [Zhang et al., 1986; Smolarkiewicz and Grell, 1992] allows for feedback to occur between the coarse-resolution domain and the nest throughout the simulation. The two domains are run simultaneously to enable this feedback. When two-way nesting is used, the portion of the coarse-resolution domain that is simulated in the nest must be discarded. The nesting ratio between domains in MM5 is generally 3:1. (Some other mesoscale meteorology models allow for user-defined nest ratios.) For example, if the coarse domain is a 36-km resolution domain, its nest will be a 12-km resolution domain. This is strictly true for a two-way nest, but is largely held as a standard for the one-way nests in MM5. As discussed in Section 3.5.1, the nest ratio restricts the number of grid points in each dimension of the nest domains to a multiple of 3, plus 1.

Figure 5.3 illustrates a sample simulation domain and its nests. Domain A is a 108-km coarse (or “parent”) domain. Domain B, which has a smaller area coverage and a higher (or finer) resolution, is a 36-km nested (or “child”) domain. Likewise, Domain C is a 12-km child domain to Domain B.

Figure 7.3. Sample domain and its nests.

7.6.6. Four-Dimensional Data Assimilation

The four-dimensional data assimilation (FDDA) scheme included in MM5 is based on Newtonian relaxation or “nudging.” Nudging is a continuous form of FDDA where artificial (non-physical) forcing functions are added to the model's prognostic equations to nudge the solutions toward either a verifying analysis or toward observations. The artificial forcing terms are scaled by a nudging coefficient that is selected so that the nudging term will not dominate the prognostic equations. In general, the nudging terms tend to be an order of magnitude smaller than the dominant terms in the prognostic equations and represent the inverse of the e-folding time of the phenomena that the observations represent.

There are two types of nudging in MM5: analysis nudging and observation nudging (“obs nudging”). Analysis nudging gently forces the model solution toward gridded fields. Analysis nudging can make use of three-dimensional analyses and some surface analyses. Obs nudging gently forces the model solution toward individual observations, with the influence of the observations spread in space and time. Obs nudging is better suited for assimilating high frequency, asynoptic data that may not otherwise be included in an analysis. Nudging in MM5 is extensively discussed in Stauffer and Seaman [1990], Stauffer et al. [1991], and Stauffer and Seaman [1994].

7.7. Meteorology Model Post-Processing

Since the output variables that are generated by an MM5 simulation are not always useful in their “raw” form, those variables must be converted into fields that are required by the chemistry and emission models. The conversion of MM5 output to useful fields for CMAQ is accomplished in the Meteorology Chemistry Interface Processor (MCIP), which is discussed in Section 2.3.1. MCIP computes variables that are useful to the subsequent models in CMAQ, and it creates an output file written with the I/O API libraries in netCDF format, which is standard in the CMAQ modeling system.

7.8. Meteorology Model Visualization and Evaluation

The MM5 output can be visualized after the post-processing in MCIP, by using either PAVE or Vis5d. If you would like to visualize MM5 prior to the processing in MCIP, several shareware tools are available for this purpose, such as Vis5d. Vis5d can be freely obtained in its original form from the University of Wisconsin; refer to http://www.ssec.wisc.edu/~billh/vis5d.html for details. A processor to reformat MM5 output for Vis5d (tovis5d) can be freely obtained from NCAR via anonymous FTP at ftp://ftp.ucar.edu in directory /mesouser/Util/vis5d. Several other graphics programs (e.g., GEMPAK, NCAR Graphics, GrADS) are available for MM5 visualization, either as freeware or for a nominal fee.

There are limited capabilities, using SAS and simple statistics available with visualization tools, to evaluate MM5 output. There are tools within the MM5 community that are used for model evaluation, but none are currently maintained by NCAR.

7.9. Model Training and User Support

There are no plans for CMAQ users to have specialized training on the use of MM5. NCAR, which maintains the official public releases of MM5, offers a periodic tutorial for setting up and running MM5. This tutorial has a limited registration to keep class sizes manageable, and it often fills several months prior to the course. NCAR charges a nominal fee for the tutorial. The tutorial notes for each class [Dudhia et al., 1998] are available on-line for no charge and can be very helpful. Note that neither UCAR (NCAR) or Penn State is obligated to provide training to users of MM5. For more information on NCAR's MM5 tutorial, refer to the MM5 home page (http://www.mmm.ucar.edu/mm5/), and click on the *User Support* section.

The CMAQ user community does not have the resources to fully support users on MM5-related questions. NCAR, which maintains the official public releases of MM5, offers limited support to the registered users via email. Because there is no charge to become a registered user or for user support (and there is no “help desk,” per se), please be patient when contacting NCAR directly for assistance. Note that neither UCAR (NCAR) or Penn State is obligated to provide any support, consulting, or assistance to users of MM5. There is also an extensive user community for MM5 that may offer informal help on specific questions via an email distribution list, which is described on the MM5 home page.

7.10. Using MM5 with CMAQ

Output files from MM5 are processed by MCIP to create the I/O API files used in the CCTM. MCIP can also be used to reduce the number of vertical layers. This is sometimes necessary as additional vertical layers greatly increase the required computer resources. The coarse MM5 grid is considered to be the “parent” grid for any nested MM5 domain. Any MCIP domain uses the MM5 domain that is at the same horizontal resolution as a “parent.” The MCIP output grid has two fewer grid cells in each direction than the MM5 domain. In addition, it is recommended that a number of boundary grid cells be eliminated for the domain used by the CCTM. The user may eliminate three or six grid cells on each edge. This must be decided when the grids are defined. Grids must be defined for MM5 and MCIP to ensure that the CCTM domain is large enough to include the area of interest. Defining grids is covered in Chapter 5. You must first identify the desired CCTM domain and resolution and work backwards to establish the MCIPand MM5 domains and resolutions.

7.11. References

Anthes, R.A., 1977. A cumulus parameterization scheme utilizing a one-dimensional cloud model. *Mon. Wea. Rev.,* 106, 270-286.

Anthes, R. A., and T. T. Warner, 1978. Development of hydrodynamic models suitable for air pollution and other mesometeorological studies. *Mon. Wea. Rev.,* 106, 1045-1078.

Arakawa, A., and V. R. Lamb, 1977. Computational design of the basic dynamical process of the UCLA general circulation model. *Methods in Computational Physics,* 17, 173-265.

Benjamin, S. G., and N. L. Seaman, 1985. A simple scheme for objective analysis in curved flow. *Mon. Wea. Rev.,* 113, 1184-1198.

Blackadar, A. K., 1979. High resolution models of the planetary boundary layer. *Advances in Environmental Science and Engineering,* 1, No. 1, Pfafflin and Ziegler, Eds., Gordon and Briech Sci. Publ., New York, 50-85.

Dudhia, J., 1993. A nonhydrostatic version of the Penn State/NCAR mesoscale model: Validation tests and simulation of an Atlantic cyclone and cold front. *Mon. Wea. Rev.,* 121, 1493-1513.

Dudhia, J., D. Gill, Y.-R. Guo, D. Hansen, K. Manning, and W. Wang, 1998. PSU/NCAR mesoscale modeling system tutorial class notes (MM5 modeling system version 2). [Available from the National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307.]

Grell, G., J. Dudhia, and D. R. Stauffer, 1995. A description of the fifth-generation Penn State/NCAR mesoscale model (MM5). NCAR Tech. Note NCAR/TN-398+STR, 117 pp. [Available from the National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307.]

Guo, Y.-R., and S. Chen, 1994. Terrain and land use for the fifth-generation Penn State/NCAR mesoscale modeling system (MM5): Program TERRAIN. NCAR Tech. Note, NCAR/TN-397+IA, 119 pp. [Available from the National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307.]

Haagenson, P., J. Dudhia, D. Stauffer, and G. Grell, 1994. The Penn State/NCAR mesoscale model (MM5) source code documentation. NCAR Tech. Note, NCAR/TN-392+STR, 200 pp. [Available from the National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307.]

Kain, J. S., and J. M. Fritsch, 1990. A one-dimensional entraining/detraining plume model. *J. Atmos. Sci.*, 47, 2784-2802.

Kain, J. S., and J. M. Fritsch, 1993. Convective parameterization for mesoscale models: The Kain-Fritsch scheme. *The representation of cumulus in mesoscale models.* K. A. Emanuel and D. J. Raymond, Eds. Meteorological Monographs. Amer. Meteor. Soc., 165-170.

Klemp, J. B., and R. B. Wilhelmson, 1978. Simulations of three dimensional convective storm dynamics. *J. Atmos. Sci.*, 35, 1070-1096.

Madala, R. V., 1981. *Finite difference techniques for vectorized fluid dynamics calculations.* Edited by D. L. Book, Springer Verlag, New York.

Manning, K., and P. Haagenson, 1992. Data ingest and objective analysis for the PSU/NCAR modeling system: Programs DATAGRID and RAWINS. NCAR Tech. Note, NCAR/TN-396+IA, 209 pp. [Available from the National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307.]

Pleim, J. E., and A. Xiu, 1995. Development and testing of a surface flux and planetary boundary layer model for application in mesoscale models. *J. Appl. Meteor.*, 34, 16 32.

Stauffer, D. R., and N. L. Seaman, 1990. Use of four-dimensional data assimilation in a limited-area mesoscale model. Part I: Experiments with synoptic-scale data. *Mon. Wea. Rev.*, 118, 1250-1277.

Stauffer, D. R., N. L. Seaman, and F. S. Binkowski, 1991. Use of four-dimensional data assimilation in a limited-area mesoscale model. Part II: Effects of data assimilation within the planetary boundary layer. *Mon. Wea. Rev.*, 119, 734-754.

Stauffer, D. R., and N. L. Seaman, 1994. Multiscale four-dimensional data assimilation. *J. Appl. Meteor.*, 33, 416-434.

Zhang, D.-L., and R. A. Anthes, 1982. A high-resolution model of the planetary boundary

layer - Sensitivity tests and comparisons with SESAME-79 data. *J. Appl. Meteor.*, 21, 1594-1609.

Zhang, D.-L., H.-R. Chang, N. L. Seaman, T. T. Warner, and J. M. Fritsch, 1986. A two-way interactive nesting procedure with variable terrain resolution. *Mon. Wea. Rev.*, 114, 1330-1339a.

Chapter 8. Developing Emissions Fields

Table of Contents

8.1. Credits and Disclaimers for Use of SMOKE 147
8.2. Obtaining SMOKE Source Code 147
8.3. System Requirements 147
8.4. Software Languages 147
8.5. Data Requirements 147
8.6. Overview of SMOKE Version 2.1 148
8.7. Emissions Model Visualization and Evaluation 151
8.8. Model Training and User Support 151
8.9. References 151

Emissions data are essential to the CMAQ model. The Sparse Matrix Operator Kernel Emissions (SMOKE) modeling system is recommended for the creation of the emissions input into CMAQ. The SMOKE model is public domain software and is being widely used in research and operational modeling applications. SMOKE is continually being updated and enhanced. An overview of the SMOKE modeling system is given below.

8.1. Credits and Disclaimers for Use of SMOKE

some text here

8.2. Obtaining SMOKE Source Code

The SMOKE source code can be downloaded from the CMAS Model Clearinghouse (http://www.cmascenter.org/html/models.html). You will need to register before you download. The files needed are the default data file, the installation script and the source code. There are also some precompiled executables for common systems.

8.3. System Requirements

SMOKE can run on many hardware platforms including SGI, Sun, Hewlett Packard, IBM and personal computers running Linux. SMOKE requires a good deal of memory and will require large amounts of disk space (this depends on the domain size).

8.4. Software Languages

SMOKE is written in Fortran 90. All of the code is compiled using the “make” utility. The run scripts are written in C-shell to accommodate the wide range of hardware platforms. A Fortran 90 compiler is required to compile and run SMOKE.

8.5. Data Requirements

Several datasets are required to run SMOKE. SMOKE requires emissions inventories and ancillary data. Both are described below.

8.5.1. Emissions Inventories

The emissions inventories that are input into SMOKE fall into six source categories, Stationary area/Nonpoint, Nonroad mobile, On-road mobile, Point, Wildfire and Biogenic sources. The inventories should contain data for the pollutant species that are required by CMAQ. The emissions inventories used in SMOKE should cover the spatial domain that CMAQ will be covering.

8.5.2. Ancillary Data

The ancillary data files are all the data files need by SMOKE except for the emissions inventories. These files include the inventory table, gridding surrogates, speciation profiles, temporal cross-reference files, report configuration files, grid description file, etc.

8.6. Overview of SMOKE Version 2.1

The latest version of SMOKE, Version 2.1, has many updated features. It includes updated versions of BEIS3 and MOBILE6.2 for biogenic and on-road mobile emissions modeling, respectively. Both of these SMOKE features integrate predicted hourly meteorology to estimate hourly emissions fluxes from these emissions sources. The latest version also has support for polar stereographic output grids. Brief descriptions of the major steps in creating emissions files follow.

8.6.1. Importing Inventory Data (Smkinven)

The first step in creating emissions files for input into CMAQ is to import the raw inventory data. The raw inventory data is usually in ASCII format. The SMOKE program Smkinven reads in this raw data and outputs an I/O API SMOKE intermediate inventory file.

8.6.2. Spatial Allocation (Grdmat)

Raw emissions inventory data is typically spatially distributed by county, road segment or a single point. The SMOKE program Grdmat creates a gridding matrix for area, mobile and point sources. This gridding matrix is then applied to the intermediate inventory file to transform the source-based inventory emissions to gridded emissions.

8.6.3. Chemical Speciation (Spcmat)

CMAQ chemical mechanisms, such as CB4, use a set of “model species” to represent atmospheric chemistry (see Chapter 5). However, the emissions inventory contains a wide variety of pollutants. The SMOKE program Spcmat creates a speciation matrix to convert the emission pollutants to the model species for the chemical mechanism used in the CMAQ model.

8.6.4. Temporal Allocation (Temporal)

The raw inventory emissions are typically annual or seasonal values. CMAQ requires hourly emissions files. The SMOKE program Temporal creates a matrix to convert the annual or seasonal values to hourly values. This is done through the use of temporal profiles. These profiles can be easily changed by the user.

8.6.5. Emissions Growth and Controls (Grwinven and Cntlmat)

Emissions growth processing creates emissions for year(s) in the future using an available dataset. The SMOKE program Cntlmat will create a growth matrix that contains the growth factors for each source and pollutant in the inventory. Next, the SMOKE program Grwinven combines this growth matrix with the inventory to create a grown emissions inventory.

Cntlmat can also be used to apply controls to the emissions inventory. These controls can be designed to change the emissions values based on regulations. Grwinven is also used to combine this control matrix with the inventory to create a controlled emissions inventory.

8.6.6. Mobile Modeling (MOBILE6)

There are three SMOKE programs to handle mobile emissions processing, Mbsetup, Premobl and Emisfac. The Mbsetup program groups the reference counties, handles the set up of the user-defined spatial and temporal averaging approach for the meteorology data, assigns the speed profiles to the sources and creates the speed summary file. The Premobl program spatially and temporally aggregates the temperature, humidity and pressure data from the meteorology files for input into Emisfac. The Emisfac program runs the MOBILE6 model to create emission factor files for all mobile sources.

8.6.7. Biogenic Modeling (BEIS3)

Biogenic processing in SMOKE is very different from the processing of the anthropogenic sources. SMOKE currently uses the BEIS version 3 model to process biogenic emissions. First, the Normbeis3 program reads in the gridded land use data. Next, the Tmpbeis3 program creates the hourly emissions. These files are then merged together in the same way as the anthropogenic sources (see Section 5.6.10).

8.6.8. Elevated Point Source Selection (Elevpoint)

The SMOKE program Elevpoint, is used to select elevated and Plume-in-Grid (PinG) sources. This is done using criteria based on emissions values, emissions rank, stack parameters and an analytical plume rise calculation.

8.6.9. Plume Rise (Laypoint)

The SMOKE program Laypoint uses gridded, hourly meteorological data and stack parameters to calculate the plume rise for all point sources emissions. Laypoint calculates the top and bottom heights of the plume and uses these heights to determine the plumes' distributions into the vertical layers it intersects. Laypoint outputs an intermediate file containing the layer fractions for each source.

8.6.10. CMAQ Input File Creation (Smkmerge and Mrggrid)

The SMOKE program Smkmerge is used to combine the speciation, gridding and temporal matrices with the inventory. In the case of point source processing, Smkmerge will also combine the layer fractions.

The SMOKE program Mrggrid is used to combine Smkmerge outputs from different source categories into a file ready for input into CMAQ.

8.6.11. SMOKE QA (Smkreport)

SMOKE contains a very powerful reporting tool called Smkreport. Smkreport can generate many types of reports used to perform quality assurance. Currently the user can create reports of emissions by source, source category code, region, road class, layer, hour, grid cell, speciation profile, gridding surrogate code, temporal profile and elevated status. Smkreport allows all of the above to be used in any combination.

8.6.12. SMOKE Summary

Figure 5.1 is a schematic of the parallel approach to emissions processing employed by SMOKE. For each SMOKE processing category (i.e., area, biogenic, mobile, and point sources), the following tasks are performed:

- read emissions inventory data files
- optionally grow emissions from the base year to the (future or past) modeled year (except biogenic sources)
- transform inventory species into chemical mechanism species defined by an AQM
- optionally apply emissions controls (except for biogenic sources)
- model the temporal distribution of the emissions, including any meteorology effects
- model the spatial distribution of the emissions;
- merge the various source categories of emissions to form input files for the AQM
- at every step of the processing, perform quality assurance on the input data and the results

Each processing category has its particular complexities and deviations from the above list; these are described in detail in CEP, [2005]. For all categories, however, most of the needed processing steps are factor-based; they are linear operations that can be represented as multiplication by matrices. Further, some of the matrices are sparse matrices (i.e., most of their entries are zeros).

SMOKE is designed to take advantage of these facts by formulating emissions modeling in terms of sparse matrix operations, which can be performed by optimized sparse matrix libraries. Specifically, the inventory emissions are arranged as a vector of emissions sorted in a particular order, with associated vectors that include characteristics about the sources such as the state/county and SCCs. SMOKE then creates matrices that apply the control, gridding, and speciation factors to the vector of emissions. In many cases, these matrices are independent from one another, and can therefore be generated in parallel and applied to the inventory in a final “merge” step, which combines the inventory emissions vector (now an hourly inventory file) with the control, speciation, and gridding matrices to create model-ready emissions. Figure 5.1 shows how the matrix approach allows for a more parallel approach to emissions processing, in which fewer steps depend on other needed steps.

Figure 8.1. Schematic of SMOKE approach to emissions processing

8.7. Emissions Model Visualization and Evaluation

The CMAQ-ready emissions files output by SMOKE can be visualized in PAVE. This allows for evaluation on a spatial and temporal scale. The reports output from Smkreport can be loaded into Excel to create plots of the data as well.

8.8. Model Training and User Support

A two and a half day SMOKE training is offered quarterly by CMAS (http://www.cmascenter.org/html/training.html). User support is also provided by CMAS through the use of the “emregional” mailing list and the Bugzilla bug reporting system.

8.9. References

Carolina Environmental Program, 2005: SMOKE v2.1 User's Manual, http://www.cep.unc.edu/empd/products/smoke/version2.1.

Chapter 9. Defining, Grids, Layers, Domains, and Chemistry

Table of Contents

9.1. Supported CMAQ Coordinate Systems 153
9.2. Horizontal Grids 154
9.3. Vertical Layers 155
9.4. Chemical Mechanism 156
9.5. References 158

The CCTM is a three-dimensional Eulerian air quality model. With an Eulerian perspective, models simulate the chemistry, transport, and other processes that affect pollutant concentrations within a specific geographical location. In this way, air parcels are transported into and out of the modeled location. In contrast to this, Lagrangian models, such as the Industrial Source Complex (ISC) dispersion model, follow the path of individual air parcels as they are transported to different geographical locations. With an Eulerian model structure, the extent of the area of interest (or domain) and the characteristics of the three-dimensional computational grid structure that represent this domain must be specified. The domain is also sometimes referred to as the grid or grid domain. The domain is divided into individual three-dimensional grid cells. A horizontal grid specification sets the x and y dimensions of each grid cell for the entire domain. All grid cells in a specified domain will have the same horizontal resolution. The vertical resolution of each grid cell depends on the vertical layer specification and the selected coordinate system. Therefore, the physical vertical extent of individual grid cells may vary over space and time.

This chapter describes how to define new horizontal grids, vertical layers, and chemical mechanisms in CMAQ. These specifications apply to multiple programs in the CMAQ model including ICON, BCON, JPROC, and CCTM. When configuring new simulations users must define the location, extent, and structure of the horizontal and vertical grids and the chemical mechanism for representing pollutant chemical transformations. CMAQ contains several default options for these parameters that can be used as templates for setting up new configurations. Before deciding to create new definitions for these, see if the existing options are sufficient for your model simulation. If a predefined choice is not appropriate, then follow the steps described in this section to create a new definition.

Once this information has been established for the horizontal grids, vertical layers, and chemical mechanisms, go to Chapter 8 to learn how to develop new model executables for running a simulation.

9.1. Supported CMAQ Coordinate Systems

The choice of coordinate system, or map projection, for CMAQ is governed by the I/O API. While the grid resolution, horizontal, and vertical extent are determined by the meteorology configuration that is input to the model, the I/O API can be used to perform grid transformations to generate meteorology inputs to CMAQ on a number of different coordinate systems. While MM5 supports the Lambert Conformal, Polar Stereographic, and Mercator coordinate systems and thus directly supports these systems for CMAQ, the I/O API can expand these choices using its grid-to-grid transformation routines. The I/O API ultimately determines which coordinate systems CMAQ supports. The I/O API supports the following coordinate systems and can be used to perform grid-to-grid transformations between them:

- Lat-Lon
- Lambert Conformal
- Mercator

- General Tangent Stereographic
- UTM
- Polar Stereographic
- Equatorial Mercator
- Transverse Mercator

While MM5 only supports a subset of these coordinate systems, the I/O API can be used to transform the MM5 meteorology into any of the systems that it supports. Therefore the limited number of projections output by MM5 is expanded through the use of the I/O API. Generally the same meteorology used to drive CMAQ is used to generate emissions for the model. The input meteorology, emissions, and CMAQ must all be configured for the same coordinate system.

9.2. Horizontal Grids

The extent of the horizontal grid used in CMAQ is limited by the size of the domain of the input meteorology. A combination of the CMAQ meteorology processor, MCIP, and the I/O API are used to prepare the input meteorology for the CCTM and contain utilities for windowing subsets of the data, performing grid transformations, and aggregating/interpolating between grid cell resolutions. Choosing the appropriate horizontal grid scale and extent for your simulation is largely dependent on the issues that you want to address through the modeling; practical consideration should also be paid to the relationship between grid size, output file size, and execution times.

In early versions of CMAQ (pre version 4.3), Fortan include files defined the horizontal configuration. The COORD.EXT file contained data statements for defining the horizontal coordinate and grid to be used for CMAQ. CMAQ used these include files at compilation to create executables that were hard-wired to a specific grid configuration. Starting with CMAQ version 4.3, CMAQ developers transitioned to a dynamic horizontal grid configuration that could be defined at execution, forgoing the need to recompile the program each time a user wanted to simulate a new grid. Horizontal grids are now configured in CMAQ through the GRIDDESC file (Section 4.1.1). CMAQ horizontal grids are selected in the “horizontal grid definition” section of the CMAQ run scripts by setting the GRIDDESC and GRID_NAME environment variables to point to an existing grid definition file and to one of the grids defined in the file, respectively. Horizontal grids can be selected from either predefined grids contained in the distributed GRIDDESC file or by adding/modifying horizontal grids through the GRIDDESC file.

9.2.1. Using predefined horizontal grids

CMAQ is distributed with a GRIDDESC file that contains definitions for two different coordinate systems and several different grids on these coordinate systems. While the tutorial dataset packaged with CMAQ only uses two of these grids (M_32_99TUT02 and M_08_99TUT02), any of these predefined grids are appropriate for setting up CMAQ simulations. For the CCTM, which requires gridded meteorology and emissions as input, the only viable grid choice is one that coincides with available input datasets. For the other meteorology-independent processors, any of the grid definitions listed in the distributed GRIDDESC file are options. To select one of the predefined horizontal grids, set the GRID_NAME environment variable in the applicable run script to the name of the grid that you would like to model.

9.2.2. Creating or modifying horizontal grids

Creating a grid in CMAQ simply involves adding a few lines of text to the GRIDDESC file. Using a combination of the file format documentation in Chapter 4 and existing grid definitions as examples, you define new grids for CMAQ by adding a coordinate and grid description to the GRIDDESC file. Set the GRID_NAME environment variable in the CMAQ run scripts to point to the name of the new grid. The most common situation for creating a new CMAQ grid definition is encountered when you receive meteorology and/or emissions data that have not yet been modeled with

CMAQ. If the meteorology data are raw MM5 outputs, by running them through MCIP, you will automatically generate a GRIDDESC file that can be input directly to both CMAQ and SMOKE. If the meteorology data are already in the CMAQ format and you do not have access to the GRIDDESC file generated by MCIP, you can determine the grid definition of the input meteorology (and emissions) by using the netCDF utility ncdump to view the header of one of the gridded binary input files.

All of the grid definition information required by CMAQ is contained in the headers of gridded meteorology and emissions netCDF input files. If you receive emissions and/or meteorology for a CMAQ simulation you can determine the coordinate and grid definitions by viewing the header of any of these files. Use a text editor to add the coordinate and grid definitions to a GRIDDESC file.

9.2.3. Tips and recommendations - Horizontal Grids

MM5 Grids

- The number of rows (NROWS) and columns (NCOLS) should be an even integer.
- The smallest grid domain must contain at least two radiosonde observations.
- Use no more than four grids in a nested grid family simulation.
- Set Delta X equal to Delta Y to maintain accuracy of finite difference schemes.
- External boundary thickness should be set to “0”.
- Horizontal grid spacing for the MM5 nested grids generally have a 3:1 ratio, although a 4:1 ratio has been employed.

CCTM Grids

- A CTM grid uses the same horizontal resolution as its MM5 parent grid.
- Minimum grid dimensions of 30 rows and 30 columns are recommended.
- External boundary thickness should be set to “1”.
- A CCTM grid should be smaller than its parent MM5 grid by at least four grid cells and preferably by six.

9.3. Vertical Layers

Two types of vertical coordinates are available in CMAQ version 4.4. For hydrostatic conditions, the time-dependent hydrostatic sigma-pressure vertical coordinate system can be used in the MM5 meteorological model. This coordinate system can also be used for non-hydrostatic conditions, such cases as nested-grid simulations, where the nested grids are too small to make the same hydrostatic assumptions used in the coarser grids of the model. The other vertical coordinate system is the time-independent reference non-hydrostatic sigma-pressure vertical coordinate system, which is used for non-hydrostatic simulations in the MM5 meteorological model.

Resolving the surface boundary layer requires high resolution near the surface for meteorological simulations. To determine mass exchange between the boundary layer and free troposphere, good resolution near the boundary layer top is preferable. In addition, different cloud parameterization may perform differently depending on the layering structure. Layer definitions should be appropriate for the topographic features of the simulation domain. Aerodynamic resistance, which influences dry deposition velocities, is a function of layer thickness and the boundary layer stability. For emissions processing, the layer thickness affects the plume rise from major stacks. The vertical extent of the emission effects is determined by the thickness of the lowest model layer for the CCTM. Although a six-layer vertical grid definition is provided with the tutorial simulation examples, there is no recommendation that it be used for regulatory applications.

In the early versions of CMAQ (pre version 4.5), Fortran include files defined the vertical layer configuration. The VGRD.EXT and COORD.EXT files contained a parameter and data statements, respectively, for defining the number of layers and sigma coordinates of the vertical layer structure to be used for CMAQ. CMAQ used these include files during compilation to build executables hard-wired with the chosen vertical layers. Starting with version 4.5, CMAQ began to use a dynamic vertical layer configuration, similar to the dynamic horizontal grid, which is specified at execution rather than at compilation. The dynamic vertical layers forgo the need to recompile CMAQ each time the user changes the vertical layer configuration between simulations.

9.3.1. Using predefined vertical layers

Wait until new release to see how they're going to do this

9.3.2. Creating or modifying vertical layers

Wait until new release to see how they're going to do this

9.3.3. Tips and recommendations - Vertical Layers

- CMAQ redefines the vertical coordinates to monotonically increase with height, a capability necessary to handle a generalized coordinate system.
- MCIP may be used to reduce the number of vertical layers by collapsing layers (no interpolation capability is provided).
- Defining the vertical layering structure for CMAQ applications is complicated, and model results are sensitive to the number and definition.
- A six-layer vertical grid definition, such as the one provided with the tutorial, is not recommended for regulatory purposes because this resolution is very coarse.
- Increasing the number of vertical layers increases the CPU time and the computational complexity.
- Computational limits arise from the Courant number limitation of vertical advection and diffusion processes. When using K-theory, a very shallow layer definition increases CPU time tremendously under the convective conditions.

9.4. Chemical Mechanism

The CMAQ modeling system accounts for chemistry in three phases: gas phase, aerosols (solid or liquid), and aqueous phase. The CMAQ modeling system's existing modules for gas phase chemistry are the Carbon Bond -IV (CB-IV), Regional Acid Deposition Model-2 (RADM2), and Statewide Air Pollution Research Center-99 (SAPRC-99) gas phase mechanisms. Several variations of the base gas-phase mechanisms, with and without aqueous and aerosol chemistry are distributed with CMAQ. With the CMAQ's modularity, it is also possible to create or modify the chemical mechanism.

Gas-phase chemical mechanisms are defined in CMAQ as a series of Fortran include files. Located in subdirectories of the $M3MODEL/include/release directory, corresponding to the mechanism name, these include files define the species, source, reaction parameters, and atmospheric processes (e.g. diffusion, deposition, advection) of the different mechanism species. CMAQ mechanism configuration is more similar to the science module configuration than to the horizontal grid or vertical layer configuration in that the mechanism is defined a compilation and results in executables that are hard-wired to a specific gas-phase mechanism. If you want to change chemical mechanisms between simulations, you must recompile a new executable that includes the desired mechanism configuration.

9.4.1. Using predefined chemical mechanisms

To select a predefined mechanism configuration in CMAQ, set the “Mechanism” variable in the build scripts to the name of one of the mechanism directories located under $M3MODEL/include/release. Table 7.1 lists the available chemical mechanisms in CMAQ version 4.5 and what is included with the each mechanism. Set the “Mechanism” variable in the CMAQ build script to the Mechanism ID in Table 7.1 to select a particular mechanism.

Table 9.1. Chemical Mechanisms

Mechanism ID	Gas-phase Mechanism			4th generation CMAQ aerosols	Aqueous Chemistry
	cb4	radm2	saprc99		
cb4	x				
cb4_ae4	x			x	
cb4_ae4_aq	x			x	x
cb4_aq	x				x
radm2		x			
radm2_ae4		x		x	
radm2_ae4_aq		x		x	x
radm2_aq		x			x
radm2_cis1		x[1]			
radm2_cis1_ae4		x[1]		x	
radm2_cis1_ae4_aq		x[1]		x	x
radm2_cis1_aq		x[1]			x
radm2_cis4		x[2]			
radm2_cis4_ae4		x[2]		x	
radm2_cis4_ae4_aq		x[2]		x	x
radm2_cis4_aq		x[2]			x
saprc99			x		
saprc99_ae4			x	x	
saprc99_ae4_aq			x	x	x
saprc99_aq			x		x

[1]RADM-2 with Carter one-product isoprene oxidation mechanism
[2]RADM-2 with Carter four-product isoprene oxidation mechanism

If you are unfamiliar with these mechanisms, refer to the detailed descriptions found in Byun and Ching [1999].

9.4.2. Creating or modifying chemical mechanisms

Creating or modifying mechanisms in CMAQ requires the use of the CMAQ chemical mechanism compiler (Chemmech) to produce the required Fortran include files. Chemmech translates an ASCII mechanism listing to the include files required by CMAQ. Like all of the CMAQ processor, Chemmech is a Fortran program that must be compiled by the user prior to use. Distributed with a Makefile for compilation and run scripts for execution, Chemmech reads a mechanism definition (mech.def) file and outputs the mechanism include files.

To modify an existing mechanism, copy the mech.def file that is contained in one of the mechanism include file directories to a new directory and modify the mechanism accordingly. Provide this modified mechanism definition file to Chemmech as input to produce the mechanism include files needed to compile CMAQ. Put these mechanism include files in a new directory under the $M3MODEL/include/release directory. To invoke this new mechanism set the "Mechanism" variable in the CMAQ build scripts to the name of the new mechanism directory and compile new executables.

To create a new mechanism for CMAQ, follow a similar procedure as above for modifying mechanisms. Use an existing mech.def file as a template to format the new mechanism for inclusion in CMAQ. After formatting the mechanism in the form of the mech.def file, provide this file as an input to Chemmech to create the required include files for CMAQ. Move the resulting include files to a new directory under $M3MODEL/include release. To invoke this new mechanism set the "Mechanism" variable in the CMAQ build scripts to the name of the new mechanism directory and compile new executables.

9.4.3. Tips and recommendations - Chemical Mechanisms

- The versions of each chemical mechanism that include both aerosols and aqueous chemistry represent the most comprehensive modeling approach.
- If you are neglecting or removing cloud processes, you can use the chemical mechanism that excludes the aqueous chemistry.
- You must use the same chemical mechanism for the CMAQ chemical transport model (CCTM) and all of the input processors that are part of the CCTM system.
- The Euler Backward Iterative (EBI) chemistry solver is mechanism dependent. When you modify a chemical mechanism you will not be able to use the default EBI solver for the new mechanism. The Rosenbrock and GEAR solvers are the only mechanism-independent choices of chemistry solvers with the CCTM.
- When adding new species to CMAQ it is important to check that the sources of these new species into the modeling domain are accounted for correctly in the mechanism include files. If you are adding these species to the domain through the emissions files, the GC_EMIS.EXT include file must contain these new species. If the new species will also enter the domain as initial and boundary conditions, check the GC_ICBC.EXT file for the presence of these species.

9.5. References